

ВЕБ-АНАЛИТИК.ИНФО

WWW.WEB-ANALITIK.INFO

**КАК МЫ «БЭКАПИМ»
СЕРВЕРЫ В AMAZON**

И ОТБИВАЕМСЯ ОТ ПИРАНИЙ

» СТР. 10

СТАРТАП

НА АВИАБИЛЕТАХ

СТР. 40 ◀



13 СЕРВИСОВ

ДЛЯ ТЕСТИРОВАНИЯ САЙТА В
РАЗНЫХ БРАУЗЕРАХ

» СТР. 15

С ИГРАМИ ТАК

ЕСЛИ СОЗДАЛ ХИТ,
ОН ТЕБЯ КОРМИТ МНОГО ЛЕТ

СТР. 62 ◀

Путеводитель в мир
Интернет-Технологий

Веб-Аналитик.ИНФО

Журнал выходит ежемесячно
и распространяется бесплатно

Издательская группа

ООО «РЕЗОНАНС.НЕТ»
www.rezonans.ru
Журнал Веб-Аналитик.ИНФО
www.web-analitik.info

Издатель и руководитель проекта
Олеся Гавриленко
alesya@rezonans.ru

Дизайн и верстка
Андрей Сиренко

Авторы
Александр Демидов
Александр Сербул
Артем Овечкин
Артур Оруджалиев
Артур Хашаев
Виталий Коробкин
Елена Гутникова
Кашмир Хилл
Линдер Кани
Максим Спиридонов
Николай Самойлов
Николай Кучумов
Ольга Гончарова
Петр Кутис
Роман Веснин
Энди Гринберг

Иллюстрации
Fotobank/Getty Images
habrahabr.ru
ig-mag.ru
topobzor.com
webdesignledger.com
webmasters.ru
www.cultofmac.com
www.flickr.com
www.forbes.ru
www.seopro.ru

Реклама в журнале и на сайте журнала
siteweb@web-analitik.info

Для пресс-релизов и
информации о пресс-конференциях
siteweb@web-analitik.info

Авторам
http://web-analitik.info/authors/

За достоверность рекламной информации
ответственность несут рекламодатели. Рек-
ламные материалы не редактируются и не кор-
ректируются. Редакция ждет ваших откликов и
писем читателей. Фотографии, рукописи и дру-
гие печатные материалы не редактируются и не
корректируются. При цитировании при перепе-
чатывании материалов журнала ссылка на сайт
www.web-analitik.info и название журнала Веб-
Аналитик.ИНФО обязательна. Полное или ча-
стичное воспроизведение материала журналов
возможно только с письменного разрешения
ООО «РЕЗОНАНС.НЕТ». Мнение редакции
журнала может не совпадать с мнением авто-
ров статей, публикуемых в журнале. Все товар-
ные знаки принадлежат их владельцам.

© ООО «РЕЗОНАНС.НЕТ»
© WEB-ANALITIK.INFO



НОВОСТНОЙ ДАЙДЖЕСТ

ХОСТИНГ

«Облако» как альтернатива традиционному
хостингу.....7

Как мы «бэкапим» серверы в Amazon и отбиваемся от
пираний.....10

ВЕБ-РАЗРАБОТКИ

13 сервисов для тестирования сайта в разных браузерах15

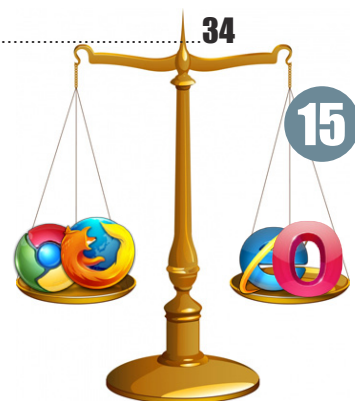
Дерево ван Эмде Боаса20

Веб-дизайн тренды 2011.....23

CMS

Что такое «асинхронная событийная модель»,
и почему сейчас она «в моде».....29

Шаблоны проектов на основе PasteScript.....34



СТАРТАП

Павел Башмаков, Stanfy Мы создали компанию, выложив свои лабораторные работы в Интернет.....	37
Стартап на авиабилетах.....	40
Стартап по-русски сегодня и завтра.....	43



ИНТЕРНЕТ

Пользователей раздражает регистрация на сайтах.....	46
Завел Facebook - забудь о privacy.....	49
Маркетинг Стива Джобса.....	51
Александр Люстик - советы и SEO-откровения.....	58
С играми так - если создал хит, он тебя кормит много лет.....	62
Как запомнить все пароли.....	68
Стереоизображение - это просто.....	69



Стив Джобс подал в отставку с поста генерального директора

Apple

Стив Джобс покинул пост генерального директора компании Apple. Об этом он сам сообщил в письме, адресованном совету директоров компании и сообществу Apple. Место Стива Джобса на посту генерального директора компании Apple занял пятидесятилетний Тим Кук (на фото), который до этого занимал должность главного операционного директора и работает в Apple уже 13 лет.



«Я всегда говорил, что если настанет день, когда я больше не смогу исполнять свои обязанности как генеральный директор Apple, я первым делом скажу об этом вам. К сожалению, такой день настал, – пишет Джобс. – Таким образом, я подаю в отставку с поста генерального директора Apple. Если совет директоров сочтет это целесообразным, то я хотел бы продолжить работу в компании в качестве председателя правления и директора».

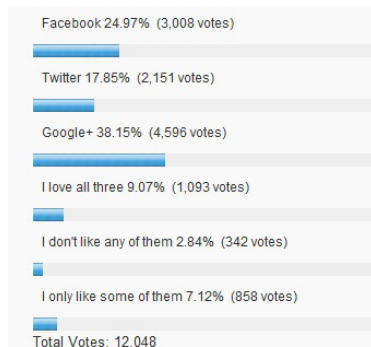
Причины своей отставки Джобс не называет, но давно известно, что у него проблемы со здоровьем.

Результаты опроса: Google+ оказался популярнее Facebook и Twitter

Среди трех социальных сетей – Facebook, Twitter и Google+ – последняя оказалась самой популярной. Такой результат опроса, проведенного среди читателей одного из крупнейших в мире ресурсов об интернете Mashable. В течении недели в голосовании приняло участие более 12 тыс. пользователей, 38,15% из которых проголосовали за Google+.



Facebook отдали предпочтение 25%, а Twitter получил 17,85% голосов.



Google+ – была запущена в конце июня 2011 года и является самой быстрорастущей социальной сетью в мире. За первых три недели существования, ее посетило около 20 млн. уникальных пользователей со всего мира.

«Яндекс» создал проблему любителям перепечатывать чужие тексты

Яндекс запустил специальную функцию, которая позволяет владельцам контента, предупредить поисковую систему о скорой публикации своих текстов. Таким образом, Яндекс будет знать, на каком именно сайте впервые появился оригинальный текст и попытается использовать это в настройке поисковых алгоритмов.

Загрузить текст можно через форму в Яндекс.Вебмастере. Необходимо загружать только оригинальные тексты, которые до сих пор не были опубликованы в интернете. Рекомендуемый минимальный объем — 2000 знаков, максимальный — 32000 знаков. Публиковать текст на сайте можно сразу после принятия заявки.

Данная возможность доступна сайтам с ТИЦ не менее 10. Яндекс обращает внимание, что не гарантирует учет заявки в работе поисковых алгоритмов.



GroupMe поможет Skype получить миллиард пользователей

Skype приобретает сервис группового общения GroupMe. Условия сделки не разглашаются, однако авторы блога Allthings D, ссылаясь на анонимный источник, называют сумму сделки - \$85 млн. Ресурс Beta Beat, в свою очередь, оценивает сумму сделки в диапазоне от 50 до \$100 млн. Прошлой осенью стартап GroupMe оценивался в \$35 млн.



Генеральный директор Skype Тони Бэйтс (Tony Bates) рассказал, что он вел переговоры с GroupMe в течение нескольких месяцев в то же время, когда Microsoft вела переговоры о покупке Skype. Он считает, что GroupMe останется отдельным брендом, а Skype будет искать пути интеграции с ним.

«Групповые сообщения – один из важнейших рынков для Skype, – сказал Бейтс. – Так же, как и Skype, GroupMe является ежедневной интерактивной формой общения. Цель Skype – получить 1 млрд пользователей. Выход на рынок мобильных телефонов создает для этого все возможности».

Мобильный сервис GroupMe позволяет создавать группы контактов, рассылать текстовые сообщения отдельным группам и устраивать бесплатные голосовые конференции.

Facebook расширил настройки конфиденциальности

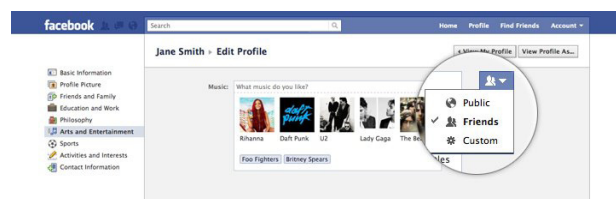
Система управления конфиденциальностью в Facebook подверглась, пожалуй, наиболее ощутимым изменениям, за всю историю существования социальной сети. Теперь пользователь сможет с высокой точностью указать, кто будет видеть его фотографии, статусы и уведомления о местоположении, говорится в официальном блоге Facebook.

Одним из самых главных изменений является то, что теперь каждый пользователь сможет утвердить или отклонить тэг, на какой-либо фотографии, прежде чем эту отметку увидят все остальные.

Запущена функция «View Profile As», которая позволяет просмотреть собственный профиль глазами одного из своих друзей.

Публикуя новый пост, можно выбрать, кому он будет виден: всем, друзьям или избранным читателям.

Всего изменений более десяти. И они коснулись двух ключевых направлений: настройки конфиденциальности в профиле и настройки, которые касаются того, как пользователь делится своим контентом с другими участниками сети. Все эти изменения новшества заработают в течение нескольких дней.



Farminers — новый бизнес-инкубатор для российских интернет-проектов

Игорь Мацанюк, известный в России как один из бывших руководителей Mail.ru Group, и Алена Владимировская — учредитель Pruffi — не так давно представили свой новый проект — бизнес-инкубатор, в котором авторы интересных идей проектов в Интернет могут получить финансовую поддержку для развития своих начинаний.

Молодым и перспективным предпринимателям будут предоставлены современные, оборудованные по последнему слову техники, офисы, а размер финансовой поддержки будет составлять до 150 тысяч долларов. Предоставляется такая финансовая помощь на срок от 3 до 6 месяцев. Кроме того, начинающие предприниматели, смогут воспользоваться экспертными оценками специалистов крупнейших проектов Рунета.

Финансирование будет осуществлять сам Игорь Мацанюк, за что он будет получать долю в проектах в размере 40%. Подобные начинания уже давно реализуются на Западе, например - Y Combinator = инкубатор Пола Грэма. Правда, его финансовая поддержка составляет до 20 тысяч долларов, а доля в проекте — до 10%.

Основатель Mail.ru Group Юрий Мильнер занялся поддержкой интернет-проектов еще зимой 2011 года, а летом 2011 года с подобной инициативой выступил и центр Digital October. Свой проект по финансовой поддержке перспективных проектов в сети есть и у поискового гиганта Рунета — Яндекс — и называется он Яндекс. Фабрика.

Как утверждают эксперты, финансовая политика нового инкубатора вполне соответствует стандартам России, а вот сроки до 6 месяцев для российского интернет рынка слишком оптимистичны. Для полноценного развития таким проектам потребуется не менее девяти месяцев.





ХОСТИНГ

«Облако» как альтернатива традиционному хостингу

Как мы «бэкапим» серверы в Amazon и отбиваемся от пираний

Автор:

Александр Демидов
demidov.tel

как альтернатива традиционному хостингу

На последнем РИФе я рассказывал об «облаке» как альтернативе традиционному хостингу (на примере Амазона).

С тех пор прошло несколько месяцев. За это время я многократно дискутировал как с ярыми противниками «облака», так и с не менее активными сторонниками.

Последний такой спор случился пару дней назад непосредственно с хостерами. И закончился (с их стороны) примерно таким выводом: «Сейчас облако в хостинге — маркетинговое зло, которое только путает людей».

Тема, как мне кажется, не просто популярная и интересная, но и очень важная. Поэтому я хотел бы обобщить свой собственный опыт (у меня есть несколько личных сайтов на виртуальном shared хостинге, периодически по знакомству присматриваю за одним дедиком, а все рабочие проекты размещены в «облаке» Амазона) и вместе с вами постараться разобраться во всех плюсах и минусах облачного хостинга по сравнению с традиционным.

Говоря в целом о традиционном хостинге, я буду подразумевать любое размещение веб-проектов — чаще VPS, solo или dedicated и чуть в меньшей степени шаред хостинг.

Под «облаком» я в общем случае буду иметь в виду IaaS (Infrastructure as a Service) — «предоставление компьютерной инфраструктуры (как правило, в форме виртуализации) как услуги на основе концепции облачных вычислений» (Wikipedia).

(Вообще, обещаю в тексте пореже употреблять разные «...aaS».)

«Облако» — звучит позитивнее, а для конечных пользователей создается хотя бы видимость понятности того, о чем идет речь. Хотя и этот термин уже «замусорен».)

Для потребителя «облако» — это любая вычислительная структура, предоставляемая как сервис по его запросу или автоматически.

С точки зрения провайдера это:

- Аппаратные ресурсы (серверы, сетевое оборудование, системы хранения данных, каналы связи и т.д.)
- Системное ПО (виртуализация)
- Управляющее ПО (биллинг, API и т.д.)

Едва ли не самый частый и самый активно использующийся посыл к переходу на «облака» вообще (и IaaS — в частности) — **уменьшение затрат и экономия**.

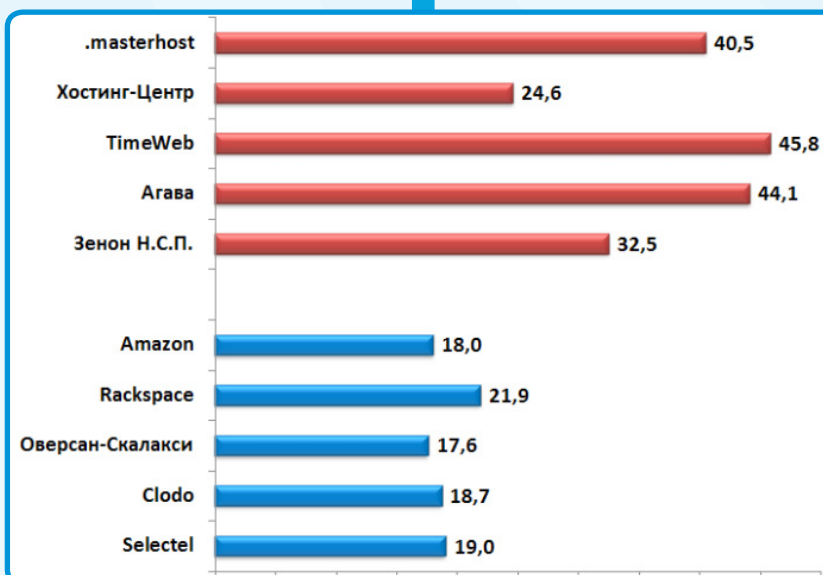
И в этом, как мне кажется, состоит очень большая и серьезная ошибка «облачных» маркетологов.

Давайте попробуем разобраться, есть ли эта самая экономия, и в чем она.

Табличка ниже — сравнение некоторой типовой конфигурации VPS «традиционных» хостеров и виртуальных машин в облаке: приблизительно 800 Mhz CPU, 512 Mb RAM. (Обычный shared хостинг в данном случае не берем в расчет, так как не сможем оценить выделенные ресурсы. Как бы ни лимитировали хостеры те или иные ресурсы, все равно обычный хостинг — это общежитие: кто первый встал — того и тапки.)

Цены — в у.е. (похожие на доллары :)).

Сравнение это не идеально, не следует его использовать как серьезный ориентир при выборе провайдера. Например, у ▶



некоторых хостеров нет строго той конфигурации, которую сравниваем мы, поэтому – выбираем ближайшую похожую.

Цель таблички – показать лишь соотношение цен. «Облако» в среднем в полтора-два раза дешевле.

Если же взять для сравнений более «взрослые» конфигурации, например, арендованных физических серверов и аналогичных по мощности виртуальных машин, то мы получим примерно похожее соотношение.

Вроде бы – дешевле, но...

Мы начинаем работать в облаке и сталкиваемся с такими (не самыми очевидными на первый взгляд) сложностями:

- Почти всегда отдельно рассчитывается стоимость трафика.

- Мы (возможно, впервые в жизни) начинаем сталкиваться с таким понятием как «расчеты по потреблению»: если наша виртуальная машина не с жесткой фиксированной ценой, то мы платим за потребленное процессорное время, использованную память; в любом случае – сталкиваемся с таким понятием как IOPS (Input-Output Operations Per Second) – и платим за это.

- При оплате «по потреблению» при резком росте нагрузки (DDoS, «хабразфлект», ошибки в разработке) возможны значительные расходы (в разы больше запланированных).

Где же реальная экономия?

На самом деле, мы не напрямую сокращаем текущие расходы, а оптимизируем несколько иных процессы.

Нет инсталляционных платежей

Для больших проектов – если речь идет о покупке собственного оборудования – больше не нужно выделять значительные средства из бюджета.

Из этого пункта автоматически следует еще одно важное преимущество «облака»:

Минимальные финансовые риски на старте нового проекта

Если вы запускаете новый стартап, то не можете заранее гарантированно знать, взлетит он или нет. «Пошел» проект – отлично, все вложения (в том числе в инфраструктуру) будут окуплены.

А если нет? И вы при этом уже потратились на серверы, софт, заплатили зарплату тем сотрудникам, которые занимались разворачиванием всей инфраструктуры?

В «облаке» вы рискуете только абонентской платой за тот период, в течение которого обслуживалась ваша виртуальная инфраструктура.

Потери в случае неудачи не столько велики (а зачастую – вообще минимальны). И они не отобьют у вас желания реализовывать новые идеи и запускать новые стартапы.)

Человеческие ресурсы для обслуживания системы

Если речь идет о большом парке машин (десятки-сотни-тысячи), то в виртуальной среде обслуживать их гораздо легче. Компания Microsoft рассказывала о собственном опыте: один системный администратор может обслуживать в «облаке» в десять (и более) раз больше машин, чем реальных физических серверов.

Экономия времени

Любые сервисные работы (добавление дисков на серверы, добавление новых машин и т.п.) в «облаке» будут выполнены быстрее.

А теперь давайте попробуем разобраться, есть ли у «облака» другие (возможно, даже более очевидные) преимущества?

Must have любого по-настоящему хорошего проекта – регулярные нагрузочные тесты (как перед запуском, так и в процессе работы – на стендах). Нагрузочное тестирование позволяет определить предел работоспособности созданного проекта именно на выбранном оборудовании и дает примерную оценку – когда потребуется масштабирование тех или иных ресурсов.

А иногда масштабирование нужно внезапно! :)

Например, живет себе в сети сайт, посвященный шахматам. И заходит на него 1-2 тыс. человек в день. И тут «внезапно» и непредсказуемо случается шахматный турнир. Посещаемость вырастает в разы, а администратор оказывается не готов к подобной нагрузке.

Оба этих проекта – и «сферический идеальный веб-сайт», и наш шахматный сервер объединяет одно: любое масштабирование потребует времени на выделение новых ресурсов, на перенос проекта на более мощное оборудование и т.п. Во время этих работ (если переезд внезапный) сайт, скорее всего, будет недоступен для посетителей. А это все – деньги владельца проекта.

Следующий момент. Рано или поздно мы упираемся в потолок: мы не можем наращивать мощность сервера неограниченно. К тому же с ростом мощности сервера его стоимость растет не пропорционально. И тут мы переходим к горизонтальному масштабированию (если поддерживает архитектура проекта) – наращиваем не мощности одного сервера, а добавляем новые машины.

И наконец, последний крайне важный момент – обратное масштабирование, чтобы ресурсы не работали вхолостую, и чтобы мы не думали о том, куда деть незадействованные серверы. Шахматный турнир идет не круглый год. Через некоторое время ресурсы, выделенные на пиковую посещаемость, оказываются невостребованными (до следующего столь же значимого события).

Масштабирование в «облаке» — простая, понятная и чрезвычайно легкая операция!

- Увеличение ресурсов виртуальной машины доступно моментально.

- Столь же моментально доступно не только вертикальное, но и горизонтальное масштабирование (можно получить практически любое количество виртуальных машин нужной конфигурации).

- После того, как отпадает необходимость в большом количестве ресурсов, мы просто отказываемся от них и перестаем за них платить.

Моментальная доступность любых ресурсов – вообще одно из важнейших преимуществ

«облака»

Живой пример, с которым мне как-то довелось столкнуться на практике – срочная необходимость переноса большого «тяжелого» проекта (который жил на отдельном выделенном физическом сервере) в другой датацентр 7-го января.

В нашей стране Новый год отмечают с размахом. И работают в новогодние каникулы лишь те проекты, которым посчастливилось не столкнуться с проблемами в тот период, когда вся страна пьет и гуляет.

Итог обзвона полутора десятков российских хостеров, которые предлагают услуги dedicated / colocation: можно разместить свой проект только на типовой стандартной конфигурации, которая уже есть в прайс-листе. И то – только по знакомству. И в лучшем случае – в течение суток. А что-то нестандартное и тем более быстрее – не получится... :(

Итог – проект перенесен в «облако» Амазона.

Напоследок хочу немного поговорить о надежности размещения проектов в «облаке».

Часто слышу такой тезис: облако надежнее.

На мой взгляд, он вводит пользователей в заблуждение и в итоге работает только против.

Не «облако надежнее», а «облако» дает гораздо больше возможностей построить надежную инфраструктуру

1. Почти все серьезные «облачные» провайдеры владеют несколькими датацентрами, что позволяет им резервировать (на своем уровне) практически все ресурсы. Для конечных пользователей это дает воз-

можность разместить свой веб-проект не на одном виртуальном сервере, а на веб-кластере, состоящем из нескольких резервирующих друг друга серверов, расположенных в разных датацентрах.

Если же нет возможности (например, дорого) постоянно держать полную копию проекта, можно использовать значительно более слабую машину, например, под бэкап данных и базы (например, slave MySQL). В случае аварии проект на резервном сервере быстро переводится в рабочий режим, а сам сервер масштабируется до нужного количества доступных ресурсов (память, CPU и т.д.)

2. Многие «облачные» провайдеры предлагают специализированные облачные хранилища для статического контента (например, Amazon S3).

Они на самом деле надежны, так вся их архитектура такова, что уже непосредственно в ней заложено резервирование на уровне датацентра.

Перенос статического контента в такое хранилище позволяет минимизировать размер и время создания бэкапов. Соответственно – минимизировать и время их разворачивания в случае какой-либо аварии.

3. Те же бэкапы – можно хранить в том же облачном хранилище. Во-первых, подобные хранилища – дешевые. Это позволяет создавать бэкапы часто и хранить их достаточно долго. Во-вторых, хранилище, которое не зависит от самих виртуальных машин, обеспечивает доступность бэкапов даже в случае самых серьезных аварий и невероятных случаях – даже если в датацентр попадает молния.

Если же проект размещается в одном датацентре на одном или нескольких физических серверах, то в случае аварии (обесточен датацентр):

■ У владельца нет доступа к данным. Вообще. Остается только ждать, когда авария будет ликвидирована.

■ Если нет возможности ждать (крайне

критична доступность сервиса) — нужно брать данные из бэкапа. Из какого? Как и куда его делать? В другой датацентр?

■ Отдельный вопрос – в какой другой датацентр переезжать во время аварии. Найти сервер нужной конфигурации (как мы уже разбирали выше) – не самая тривиальная задача. Требуемая времени.

■ Переехали. Надо менять DNS. Это – опять время. В случае использования «облака», у которого есть несколько датацентров, можно просто «перевесить» прежний IP на новую машину.

Кстати, именно дополнительные сервисы зачастую являются одним из факторов, по которым в итоге выбирается «облачный» провайдер.

Возможность размещения в разных датацентрах, множество готовых образов операционных систем, обеспечение доступности в SLA, гибкие конфигурации дисков, «облачные» базы данных, «облачный» кэш данных, конфигурирование файрволла через веб-интерфейс для групп машин, создание копий виртуальных машин и т.д. и т.п.

Масса возможностей для «облачных» провайдеров для конкуренции и привлечения клиентов.)

Закончу я свой пост цитатой Брета Тейлора, ныне — технического директора Facebook, а немного ранее — сооснователя FriendFeed: «Когда мы только начинали работу над стартапом (FriendFeed), нам нужно было решить, покупать собственные серверы или же выбрать одного из «облачных» хостинг-провайдеров – таких как Amazon (AWS). Мы выбрали первое – решили приобрести собственные серверы. Оглядываясь назад, я думаю, что это было большой ошибкой».

+	Не является альтернативой хостингу за 200-300 руб./мес.
	Затраты (время) на обучение сотрудников специфике конкретного сервиса
	Ограничения инфраструктуры (аппаратная часть, специфичное ПО)
	Сложность расчетов «по потреблению»
-	Экономия за счет возможности планирования ресурсов
	Экономия и отсутствие рисков, связанные с вложениями в инфраструктуру
	Моментальное вертикальное и горизонтальное масштабирование
	Удобство администрирования
	Экономия времени
	Дополнительные сервисы

Автор:

Александр Сербул
[www.1c-bitrix.ru/
 blog/alexander_
 serbul/](http://www.1c-bitrix.ru/blog/alexander_serbul/)



Как мы «бэкапим» серверы в Amazon и отбиваемся от пирани



Многие владельцы веб-проектов, размещенных в популярном облачном провайдере Amazon, наверняка задумываются о том, как создать эффективную и надежную конфигурацию веб-решения, как проводить резервное копирование машин, как действовать в случае коллапса датацентра, в котором размещены ваши серверы. К сожалению, официальная документация облачного хостинга «несколько» скупа на тему надежности и внутренней реализации веб-сервисов — поэтому приходится полагаться на собственный боевой опыт эксплуатации и армейскую смекалку.

Ситуацию усугубляет наблюдаемое ныне противостояние распространенного заблуждения, что в облаке все очень надежно и можно крепко спать: жесткие диски «из титана», сетевой трафик течет «по золотым проводам — [с рекомендациями облачного провайдера](#) на тему «мы предоставляем сервисы достаточной надежности и быстрые каналы между ДЦ, а задача архитектора проекта — комбинировать наши технологии, доводя девяточки справа до нужного количества».

В статье хочу поделиться нашими наблюдениями заведением сервисов в течение полугода, архитектурными решениями на EBS-дисках для обеспечения надежности и производительности веб-проектов и конкретными техниками резервного копирования, которые мы используем в повседневной работе.

Надежность EBS-дисков — RAID1 или «хуже»?

Если честно, из официальной документации Amazon по EBS-дискам я так и не понял, надежнее ли они RAID1 или нет —

всякие разговоры про функциональную зависимость объема диска, времени последнего снапшота диска, числа волос в бороде волшебника и фазы луны не внушают оптимизм:

«The durability of your volume depends both on the size of your volume and the percentage of the data that has changed since your last snapshot. As an example, volumes that operate with 20 GB or less of modified data since their most recent Amazon EBS snapshot can expect an annual failure rate (AFR) of between 0.1% – 0.5%, where failure refers to a complete loss of the volume»

Во время недавней аварии в европейском ДЦ Амазона у нас из софтверного RAID1 вылетели сразу 2 диска из-за отключения питания в датацентре. В сети также иногда появляются жутковатые сообщения о том, что диск EBS внезапно «сломался», клиенту прислали письмо на тему «ваши данные потерялись, к сожалению» и официально рекомендуется [сделывать снапшот](#) как раз за минуту перед отказом диска :) Такое впечатление, что механизм снапшотов появился именно для «подстраховки» от выхода из строя не очень надежных EBS-дисков.

Публичные [виртуальные машины](#), которые используются для создания собственных приватных образов, имеют только один EBS-диск, который не в рейде и на котором корневой раздел. А загрузить машину с корневым разделом на софтверном рейде — мягко сказать, без напильника непросто и требует времени.

Тем не менее, за полгода в Амазоне у нас не вылетел ни один диск (у нас их около 50), не считая отключения питания после удара молнии, что позволяет относиться к ним как к более-менее надежным «RAID1» дискам (тук, тук; репликация на одно устройство, расположенное в том же датацентре), уступающим в надежности [s3](#) (реплицируется дополнительно на 2 устройства в разных датацентрах, и вообще это не блочное устройство), но в 10 раз более надежным чем обычные жесткие диски. ►

Софтверный рейд и невысокая производительность EBS-дисков

После миграции в Амазон в начале этого года мы столкнулись с достаточно невысокой производительностью EBS-дисков, что было особенно заметно на машине с MySQL. «Подсмотрев» идею в [RDS](#) (Амазон автоматически, в зависимости от суммарного объема EBS-дисков, размещают информацию базы данных на софтверном рейде) и использовав армейскую смекалку мы перенесли данные на созданный из EBS-дисков софтверный RAID10 (mdadm в Linux):

```
cat /proc/mdstat
Personalities : [raid10]
md0 : active raid10 sdo[0] sdj[8](S) sdi[7] sdh[6]
sdg[5] sdn[4] sdm[3] sd1[2] sdk[1]
629145344 blocks 64K chunks 2 near-copies [8/8]
[UUUUUUUU]
```

Ах, да, spare-disk, как вы видите, мы используем :) «sdj[8](S)». Решили все таки поставить в рейды для увеличения уровня отказоустойчивости. После миграции данных на рейды проблемы с производительностью дисковых подсистем на машинах перестали беспокоить.

Кластерные файловые системы и EBS-диски

Т.к. наш основной сайт (www.1c-bitrix.ru) работает в веб-кластерной конфигурации: две веб-машины, две машины с базой данных (мастер+слейв) — за балансировщиком (nginx), мы очень, очень хотели использовать кластерную файловую систему типа OCFS2 или GFS2. Но EBS-диски оказалось нельзя одновременно [смонтировать](#) на несколько виртуальных машин. Поэтому мы используем для кластеризации контента утилиту [csync2](#), форсированную надстройкой для «быстрой» синхронизации часто обновляемых тяжелых папок на базе [inotify](#) (если интересно, расскажем подробнее).

Бэкапим EBS-диски

Диски виртуальной машины совсем несложно бэкапить, т.к. именно для этого создан инструмент создания снимков блочного устройства:

```
ec2-create-snapshot vol-a5rtbvwe
```

Операция стартует очень быстро (секунды), а загрузка снимка в s3 продолжается определенное время, в зависимости от объема диска и загруженности Amazon — до десятков минут.

Мы взрослые люди и понимаем, что целостный снимок диска может быть если:

1. Диск предварительно отмонтирован:
«umount -d /dev/sdh».
2. Сервер предварительно остановлен.
3. Файловая система «заморожена». Хотя в этом случае нужно еще приложениям сказать, чтобы сбросили данные на диск.

Иначе мы получаем снимок диска «как бы после внезапного выключения питания» и диск нужно будет полечить (fsck) при загрузке машины — что, благодаря, современным журналируемым файловым системам (ext3, ext4, xfs и др.) происходит незаметно и часто быстро за счет проигрывания журнала. Мы тестировали создание снимка диска с «живой» (без размонтирования) ext3 и xfs — после некоторой паузы для проигрывания журнала диск (созданный из снимка) монтируется и работает.

«Замораживание» и «отмораживание» файловой системы

В качестве файловой системы, поддерживающей «приостановку», мы выбрали xfs. Эта ФС рекомендуется Amazon и используется утилитами создания целостного снимка типа ec2-consistent-backup:

```
xfs_freeze -f | -u mount-point
```

Действительно, если после «xfs_freeze -f» сделать снимок диска, он монтируется без проигрывания журнала очень быстро.

В CentOS6 (и видимо в других дистрибутивах с «обновленным» ядром) стало возможным «фризить» также дефолтные файловые системы типа ext3/ext4 командой fsfreeze. Мы планируем хорошенько потестировать эту утилиту в контексте создания целостных снимков EBS-дисков.

Бэкап софтверного рейда

Если мыслить логически, то нужно а) «остановить» файловую систему рейда, б) запустить создание снимка для КАЖДОГО физического диска рейда, в) «отпустить» файловую систему рейда. К сожалению, в документации к API Amazon нет информации, сколько времени должно пройти между б) и в). Однако опытным путем было найдено (и аналогично делается в данной ►

утилите), что достаточно получить идентификаторы снапшотов из вызова API для каждого диска рейда и файловую систему рейда можно «разморозить» — и при сборке рейда он подхватится без ребилдинга. Иначе — начнет ребилдиться. Вот так сделали мы:

```
bxc_exec («/usr/bin/ssh remote_xfs_freezer@
hostname».escapeshellarg («sudo /usr/sbin/xfs_
freeze -f /xfs_raid_path»));
...
ec2-create-snapshot - для каждого диска рейда
...
bxc_exec («/usr/bin/ssh remote_xfs_freezer@
hostname».escapeshellarg («sudo /usr/sbin/xfs_
freeze -u /xfs_raid_path»));
...
ec2-create-volume --snapshot id_of_snapshot
...
ec2-attach-volume volume_id -i instance_id -d
device
...
mdadm --assemble /dev/mdN /dev/sd[a-z]
```

Снепшот MySQL?

Хотя можно одной рукой сбросить данные MySQL на диск: FLUSH TABLES WITH READ LOCK, левой ногой зафризить файловую систему: xfs_freeze -f /path, а носом нажать Enter для выполнения API вызова снятия с дисков снапшота: ec2-create-snapshot... нас пока полностью устраивает асинхронная репликация MySQL в другой ДЦ :). Интересным предоставляется создание снапшотов базы с использованием xtrabackup — без вышеописанных танцев с бубнами.

Можно бэкапить всю машину сразу!

Если отдельно бэкапить диски, группы дисков в рейдах, то, при наличии конфига — процесс работает стабильно и прозрачно. Однако, если вам нужно иногда (например, в ДЦ ударила на выходных молния :)) переезжать между AZ (датацентрами) Amazon — проще ввести уровень абстракции выше ишников инстансов, снапшотов и дисков и оперировать категориями ролей и образов (AMI — [Amazon Machine Image](#)) машин. Через API можно назначать объектам Amazon [тэги](#) и фильтровать по ним выборки.

Мы определили роли: веб-машина, СУБД, балансировщик, машина мониторинга и наши скрипты бэкапов работают примерно так:

1. [Найди машину](#) с тегом «СУБД» и получи ее ИД
2. «Заморозь» диски машины
3. [Сделай снапшот](#) дисков. Задай снапшоту тэг «СУБД».
4. «Разморозь» диски машины

5. Для очистки устаревших снапшотов можно [сделать выборку](#) по тэгу и времени и удалить лишние.

Однако, гораздо проще [бэкапить работающую виртуальную машину в образ](#) (AMI image). При этом в образе сохраняются ее настройки и пути к создаваемым снапшотам, из которых будут созданы диски машины, присоединяемые к ней при старте:

```
ec2-create-image id_of_instance [--no-reboot]
...
ec2-run-instances ami_image_id
```

Почувствуйте разницу! Я создаю образ с машины с рейдом10 из 9 дисков — одной командой. И не парюсь с ишниками ее дисков и снапшотов — вся необходимая для воссоздания машины информация записывается в образ AMI автоматически. Единственный минус — неясно, на сколько времени нужно лочить диски/рейды работающей машины, чтобы создать из нее образ AMI. Опытно нашли, что достаточно 2-5 секунд — если меньше, то диски/рейды начинают восстанавливаться при создании новой машины. Также очень удобно то, что поднять машину из бэкапа в образ (AMI) можно в другом датацентре (AZ).

Сейчас мы создаем два раза в сутки бэкап в образ AMI всех машин нашей кластерной конфигурации (сотни гигабайт). Очистка «устаревших» образов, вместе с их снапшотами производится скриптом:

```
function bxc_delete_old_instance_backups(
AmazonEC2 $ec2, $instanceRole = '', $bkpsMaxCount = 3) {
    if (!$instanceRole || !is_object($ec2) ||
    $bkpsMaxCount < 1 ) {
        bxc_log_error («Bad input params: «. __
        FUNCTION__»);
        return false;
    }
    $response = $ec2->describe_images(array(
        'Filter' => array(
            array('Name' => 'tag:Role', 'value'
            => 'Image:'. $instanceRole),
            array('Name' => 'state', 'value' =>
            'available'),
        )
    ));
    if ( !$response->isOk() ) {
        bxc_log_amazon_error($response, __
        FUNCTION__ .»:»: __LINE__);
        return false;
    }
    if ( isset($response->body->imagesSet->item) ) {
        $images = array();
        $snaps = array();
        foreach ( $response->body->imagesSet->item as $image ) { ►
```

```

        $images[ (string) $image->name ] =
(string) $image->imageId;
        if ( isset($image-
>blockDeviceMapping->item) ) {
            foreach ( $image-
>blockDeviceMapping->item as $disk ) {
                $snaps[ (string) $image-
>name ][] = $disk->ebs->snapshotId;
            }
        }
    }
    if ($images) {
        krsort($images);
    } else {
        bxc_log(«No images with
tag:Role=Image:$instanceRole»);
        return true;
    }
    if ( count($images) <= $bkpsMaxCount ) {
        bxc_log(«Nothing to delete»);
        return true;
    }
    $imagesToDelete = array_slice($images,
$bkpsMaxCount, count($images)-$bkpsMaxCount,
true);
    foreach ($imagesToDelete as
$imageName=>$imageId ) {
        $r = $ec2->deregister_
image($imageId);
        if ( $r->isOk() ) {
            bxc_log(«Image deleted ok:
«. $imageId);
            sleep(5); //Important. wait to ami
really be deleted ;-)
            foreach ( $snaps[$imageName] as
$snapId ) {
                $rr = $ec2->delete_snapshot(
$snapId );
                if ( $rr->isOk() ) {
                    bxc_log(«Snapshot
deleted ok: «. $snapId);
                } else {
                    bxc_log_amazon_
error($rr, __FUNCTION__);
                    bxc_log_error(«Cannot
delete snapshot: «. $snapId);
                }
            }
        } else {
            bxc_log_amazon_error($r, __
FUNCTION__);
            bxc_log_error(«Cannot delete
image: «. $imageId);
        }
    }
    return true;
} else {

```

```

        bxc_log(«No images with
tag:Role=Image:$instanceRole»);
        return true;
    }
    return false;
}

```

Админка, консоль или API

Amazon AWS имеют неплохую админку, однако многие вещи можно сделать только через API. Для Linux можно установить инструменты командной строки для работы с API Amazon, которые довольно хорошо документированы. Мы используем такие инструменты:

```

ls /opt/aws/
AutoScaling-1.0.39.0
ElasticLoadBalancing-1.0.14.3
ec2-api-tools-1.4.4.1
env-set.sh

```

Последний скрипт устанавливает переменные окружения для работы инструментов. Обратите внимание, практически каждый веб-сервис Amazon имеет свой набор утилит, которые нужно скачать и установить отдельно. Для управления машинами, например, документация находится [тут](#), а утилиты командной строки — [тут](#).

Пока неплохо себя показывает в работе официальный SDK для работы с API Amazon [для PHP](#) (вышеприведенный скрипт использует именно эту библиотеку).

Выводы

С EBS-дисками Amazon можно эффективно работать, объединив их в софтверный рейд. К сожалению, нельзя замонтировать EBS-диск к нескольким виртуальным машинам, для работы с кластерной файловой системой типа OCFS2, GFS2. Бэкапы как дисков, так и рейдов EBS делаются несложно, необходимо только понять основные принципы и обойти подводные камни. Удобно бэкапить машины целиком, а не по отдельному диску — это позволит быстро развернуть конфигурацию в соседнем датацентре если в ваш ДЦ ударит молния или врежется самолет. Всем удачи! ■



ВЕБ-РАЗРАБОТКИ

**13 сервисов для тестирования
сайта в разных браузерах**

Дерево ван Эмде Боаса

Веб-дизайн тренды 2011

Автор:

Александр Сербул
topobzor.com

13 сервисов для тестирования сайта в разных браузерах

Одно из стандартных требований заказчика к разработке качественного веб-сайта – это кроссбраузерность. Поэтому каждый веб-разработчик рано или поздно сталкивается с вопросом тестирования сделанного веб-сайта в разных браузерах.

«Кроссбраузерность — свойство сайта отображаться и работать во всех популярных браузерах идентично. Под идентичностью понимается отсутствие развалов верстки и способность отображать материал с одинаковой степенью читабельности. Понятие «кроссбраузерность» очень

часто путают с попиксельным соответствием, что на самом деле является разными понятиями». (Википедия)

Есть множество разных способов проверить ваш сайт в разных браузерах и в разных операционных системах. Например, онлайн-сервисы, локальные приложения, установка нескольких браузеров на свой компьютер и т.д. Есть платные и бесплатные сервисы. В данном обзоре мы обратим ваше внимание на наиболее оптимальные бесплатные и платные сервисы проверки вашего сайта в разных браузерах и на разных операционных системах.

1. Browsershots (бесплатно/платно)

Browsershots - это веб-сервис, который делает скриншоты вашего сайта в разных операционных системах и браузерах (всего доступно **65 браузеров**). Это удобный, хоть и довольно медленный способ проверить свой сайт сразу во многих браузерах. Когда вы вводите адрес вашего сайта в строку для проверки, он ставится в очередь на тестирование. После этого скриншоты будут появляться на итоговой странице по очереди. Это может занять от 5 минут до 2 часов. В бесплатной версии все происходит довольно медленно. Кро-



Рис. 1

ме того, одна сессия (запрос) может длиться только 30 минут.

Следует обратить внимание, что данный сайт имеет многоязычный интерфейс (в том числе – русский и украинский.)

Однако хоть данный сервис заявлен как бесплатный, в нем есть и платные услуги. За 29.95 у.е. в месяц вы получите так называемую “приоритетную обработку”: скриншоты будут появляться намного быстрее, чем в бесплатной версии, сможете получить от 30 до 50 скриншотов всего за 5 минут. Кроме того, ваши скриншоты будете видеть только вы, тогда как в бесплатной версии они в итоге выкладываются в общую ленту (рис. 1).

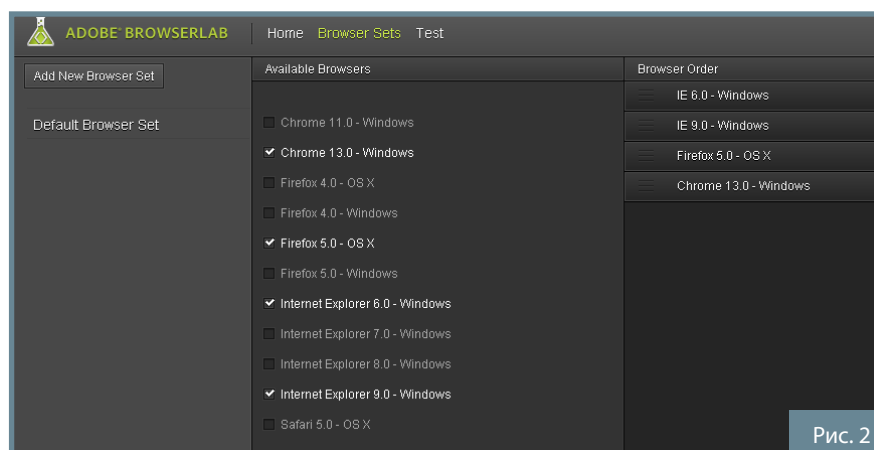


Рис. 2



Рис. 3

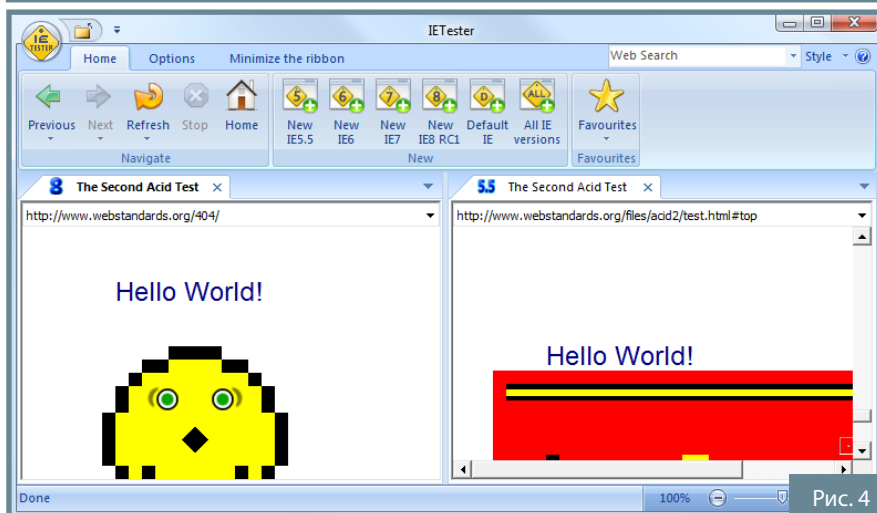


Рис. 4



Рис. 5



Рис. 6

2. Adobe BrowserLab

(бесплатно)

Adobe BrowserLab – бесплатный онлайн сервис от **Adobe CS**. Чтобы добраться до этого сервиса нужно иметь аккаунт Adobe ID на сайте Adobe.com. Потом все достаточно просто – выбираете нужные браузеры, операционные системы и получаете скриншоты. Если вы пользуетесь для веб-разработки **Adobe Dreamweaver**, то **Adobe BrowseLab** интегрируется с Adobe Dreamweaver, начиная с версии CS4 (рис. 2).

3. IE NetRenderer

(бесплатно)

IE NetRender – бесплатный онлайн-сервис только для браузера **Internet Explorer**, который дает возможность проверить ваш сайт под браузером Internet Explorer версий: 9, 8, 7, 6, 5.5.

Тут все просто: вводите адрес своего сайта, выбираете версию IE и тут же получаете скриншот (рис. 3).

4. IE Tester

(бесплатно)

IETester – это абсолютно бесплатное приложение только для браузера **Internet Explorer**, которое позволяет локально на вашем компьютере проццизматривать сайт в разных версиях Internet Explorer (версии: 10, 9, 8, 7, 6, 5.5) под операционными системами Windows 7, Vista и XP.

Для этого нужно скачать **IE Tester** и установить у себя на компьютере. Найти последнюю версию IE Tester можно тут: <http://www.my-debugbar.com/wiki/IETester/HomePage> (рис. 4). ▶

5. BrowserCam (платно)

BrowserCam – очень мощный платный онлайн-сервис для тестирования сайтов под любым браузером и любой платформой. Одна из главных «фишек» данного сервиса в том, что на нем предоставляется удаленный доступ (через VNC) к компьютерам с уже установленными ОС и браузерами, в которых вы можете тестировать свой сайт. Кроме того есть такая важная опция как тестирование под iPhone OS, Android, Blackberry, Windows Mobile. Можно также тестировать е-мейлы на разных устройствах, что очень важно для рассылочных компаний.

Цены стартуют от 19,95 у.е. в день в зависимости от функционала (рис. 5).

Рис. 7

6. Cross Browser Testing (платно)

CrossBrowserTesting – также очень функциональный платный сервис, где вы можете проверить за несколько минут сайт в разных браузерах, в разных операционных системах, на разных устройствах (поддерживается более **100 вариантов проверок**, в том числе тестирование под Android, iPad, iPhone).

Ценовая политика выгодней и гибче, чем у предыдущего сервиса BrowserCam: базовая цена стартует от 29,95 у.е. в неделю. Кроме того, дается бесплатный тестовый триальный период в 7 дней на каждый из трех пакетов, правда с несколько ограниченной квотой, нежели в платной версии (рис. 6).

Рис. 8

предоставления скриншотов, заодно проверяет валидность html и css кода. Особенность его в том, что он в основном ориентирован на тестирование е-мейлов в разных браузерах и на разных устройствах. Есть возможность протестировать е-мейл в 30 разных почтовых клиентах и другие важные функции, такие как аналитика для е-мейлов.

Платная версия сервиса предполагает три пакета: базовый (49 у.е. в месяц), плюс (99 у.е. в месяц), премиум (299 у.е.

в месяц). Бесплатная триал-версия действует 7 дней (рис. 7).

7. Litmus (платно/бесплатно)

Litmus – онлайн-сервис, который предоставляет возможность тестирования сайта на разных браузерах. Кроме

8. CloudTesting (платно)

CloudTesting – платный онлайн-сервис, который предоставляет возможность функционального тестирования кроссбраузерности сайта. Вы записываете ▶



Рис. 9



Рис. 10

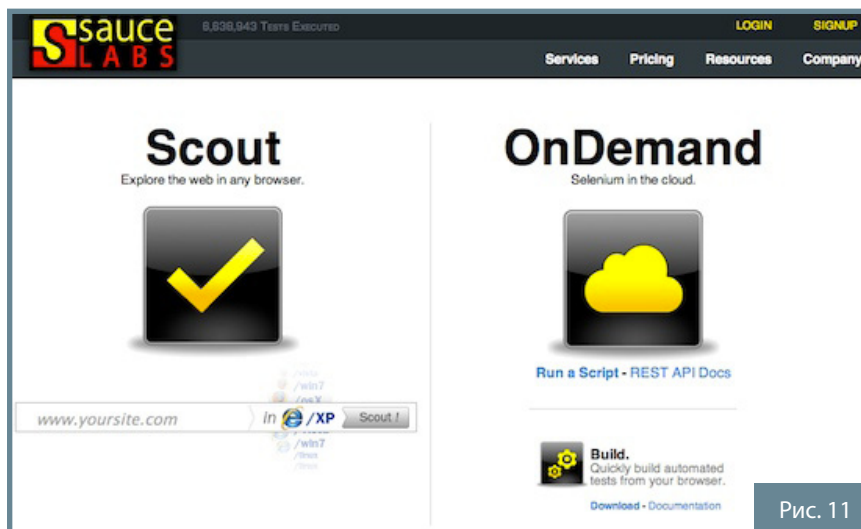


Рис. 11

сценарий поведения пользователя с вашим браузером и Selenium IDE, загружаете его, после чего **Cloud Testing** будет выполнять этот сценарий в нескольких браузерах и операционных системах. Затем он предоставляет скриншоты.

Ценовая политика стартует от 99 у.е. в месяц (рис. 8).

9. Mogotest (платно)

Mogotest – платный онлайн-сервис, который предоставляет возможность полного тестирования кроссбраузерности, в том числе и для частных страниц. Предоставляется API, поэтому возможна интеграция в существующие инструменты и рабочие процессы.

Ценовая политика стартует от 15 у.е. в месяц для физических лиц. При этом есть возможность бесплатно тестировать выбранный пакет (триальная версия) **14 дней** (рис. 9).

10. Multi-Browser Viewer (платно)

Multi-Browser Viewer – платное приложение, которое работает с десктопными и мобильными браузерами и включает в себя **26** виртуализируемых браузеров, 5 мобильных браузеров (в том числе iPhone и iPad) и **61** скриншот-браузер (с их помощью можно видеть, как визуализируются страницы, но нельзя видеть результат взаимодействия пользователя с сайтом).

Ценовая политика Multi-Browser Viewer составляет 139,95 у.е. за однопользовательскую лицензию и включает один год использования продуктов и обновлений. Кроме того доступна бесплатная триал-версия программы (14 дней) (рис. 10).

11. Sauce

Labs

(бесплатно/
платно)

Sauce Labs – онлайн-сервис, который предоставляет доступ ко множеству браузеров в разных ОС и устанавливает соединение вашего браузера с настроенной виртуальной машиной.

Сервис предоставляет платные пакеты (цены стартуют от 49 у.е. в месяц), кроме того, есть бесплатная квота на тестирование: 200 минут в месяц и позволяет создавать тесты автоматизированного тестирования в браузерах (используется Selenium) (рис. 11).

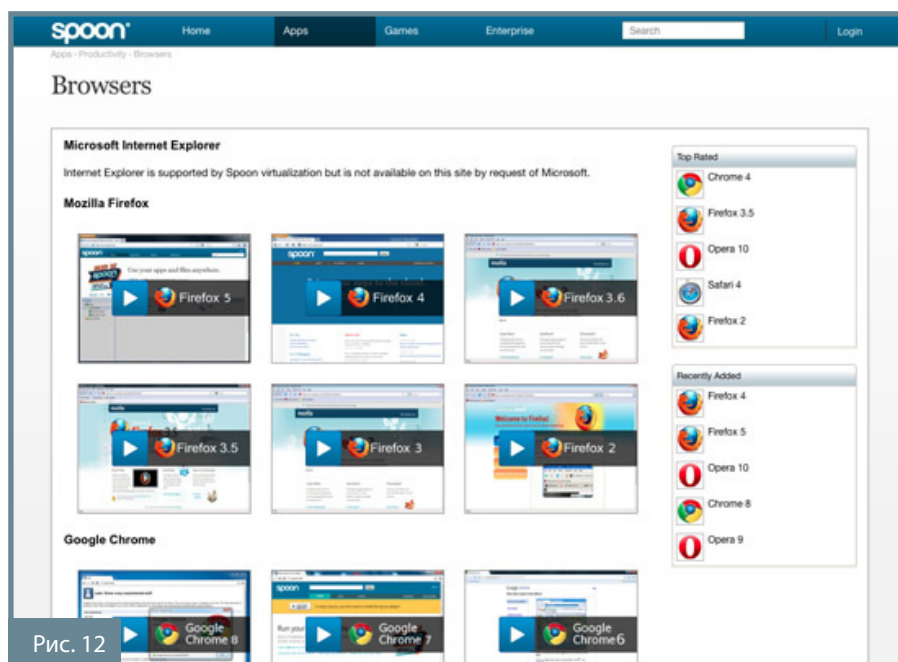


Рис. 12

телей **Windows**. Доступны разные версии браузеров Firefox, Chrome, Safari, Opera. Технически, Spoon поддерживает виртуализацию IE, но эта возможность отключена по требованию Microsoft (рис 12).

который обеспечивает автоматизацию тестирования кроссбраузерности. Он автоматически определяет различия в отображении страниц браузерами, тем самым упрощая процесс тестирования.

Бесплатная версия включает в себя довольно ограниченное число браузеров и низкое разрешение. Платные пакеты стартуют, начиная от 39 у.е. за 14 дней и от 49 у.е. до 99 у.е. за месячную подписку (рис. 13).

12. Spoon

(бесплатно)

Spoon – это онлайн-сервис эмуляции программ. Предоставляется бесплатная возможность запуска **Firefox**, **Chrome**, **Opera** и **Safari** для пользова-

13. Browsera

(бесплатно/платно)

Browsera – это онлайн-сервис, кото-

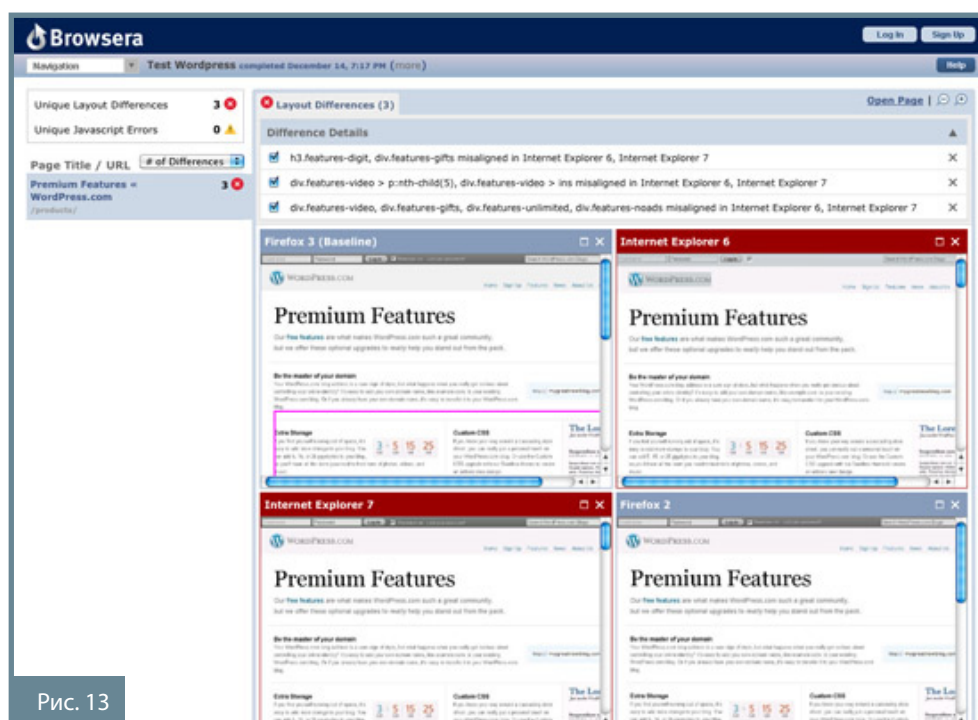
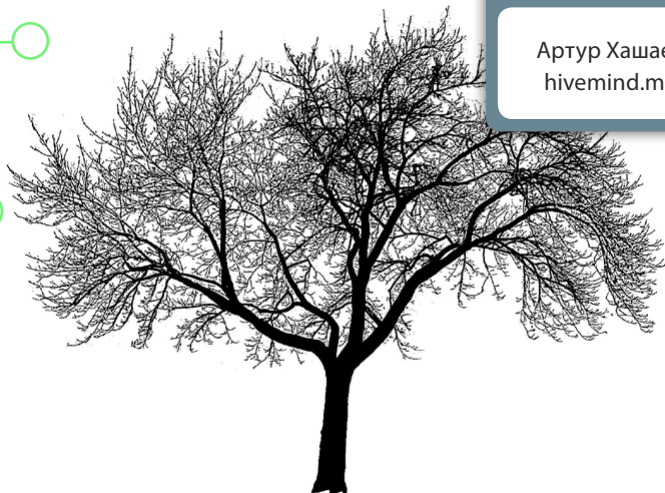


Рис. 13

Дерево ван Эмде Боаса

Автор:

Артур Хашаев
hivemind.me

Сегодня я расскажу вам об одной интересной структуре данных, про которую слышали лишь немногие и про которую очень незаслуженно мало написано в рунете, да и в англоязычной информации, в общем-то, тоже негусто. Решено было исправить ситуацию и поделиться с общественностью в доступной форме этой достаточно экзотической структурой данных.

Дерево ван Эмде Боаса (van Emde Boas tree) — ассоциативный массив, который позволяет хранить целые числа в диапазоне $[0; U]$, где $U = 2^k$, проще говоря, числа, состоящие не более чем из k бит. Казалось бы, зачем нужно еще какое-то дерево, да еще позволяющее хранить только целые числа, когда существует множество различных сбалансированных двоичных деревьев поиска, позволяющих выполнять операции вставки, удаления и прочие за $O(\log n)$, где n — количество элементов в дереве?

Главная особенность этой структуры — выполнение всех операций за время $O(\log(\log(U)))$ независимо от количества хранящихся в ней элементов.

Собственно, вот небольшой список поддерживаемых операций:

- `Insert(x)` — Вставка числа в дерево
- `Remove(x)` — Удаление числа
- `GetMin()`, `GetMax()` — Нахождение минимума и максимума в дереве
- `Find(x)` — Поиск числа в дереве
- `FindNext(x)` — Поиск следующего числа после x , которое содержится в дереве
- `FindPrevious(x)` — Поиск предшествующего x числа

При этом, конечно, можно связать с каждым из чисел какую-нибудь информацию, скажем, мы будем хранить в дереве номера каких-то товаров и по номеру сможем узнавать имя товара, то есть `Find(x)` будет возвращать не просто `True`/

`False` — существует ли товар или нет, а имя товара с номером x . Мы рассмотрим реализацию, в которой хранятся просто числа, без дополнительной информации. Прикрыть же эту фишку не составит никакого труда.

Итак, приступим к построению нашей структуры, причем строить ее будем рекурсивно.

Пусть k -дерево будет хранить числа в интервале $[0; 2^k)$, то есть, состоящие из k бит. При этом само число k будет степенью двойки, позже будет понятно, зачем нам необходимо такое условие. Тогда очевидно, как построить 1-дерево, оно будет лишь хранить информацию о том, существуют ли числа 0 или 1 в нем или нет.

Теперь построим k -дерево. В нем будет храниться:

- $2^{k/2}$ штук $(k/2)$ -деревьев
- Вспомогательное $(k/2)$ -дерево
- Минимальное и максимальное число, содержащееся в этом дереве (если оно не является пустым)

Становится непонятно, зачем нам это все нужно?

Представим теперь, что мы пытаемся вставить число x , состоящее из k бит в k -дерево, содержащее поддеревья

$$\text{binary}(x) = \underbrace{0101}_{\text{high}(x)} \underbrace{1101}_{\text{low}(x)}$$

`children[0..2k/2 - 1]`. Например, пусть $x = 93$. Посмотрим на его двоичное представление:

Разобьем наше число на две равных части `high` и `low`, каждая по $(k/2)$ бит. Вспоминаем, что в нашем дереве содержится $2^{k/2}$ деревьев, которые хранят числа из $(k/2)$ бит. А теперь возьмем из всех этих деревьев `high`-тое по счету (т. е. `children[high]`) и рекурсивно вставим в него число `low`.

Таким образом получаем простой алгоритм вставки и поиска в дереве T числа x , псевдокод:

```
insert(T, x):
```

```
    if T.k == 1: ▶
```

```
T.exist[x] = True # обрабатываем
отдельно случай вставки в 1-дерево
```

```
else:
```

```
insert(T.children[high(x)], low(x))
```

```
find(T, x):
```

```
if T.k == 1:
```

```
return T.exist[x]
```

```
else:
```

```
return find(T.children[high(x)], low(x))
```

Оценим асимптотически время работы этих функций. Пусть $T(k)$ — количество операций, требующихся для вставки/поиска числа в k -дереве. Для вставки/поиска в 1-дереве нам необходимо $O(1)$ операций, получаем, что $T(1) = O(1)$. Если же мы рассмотрим k -дерево ($k > 1$), то получим, что распил числа с помощью битовых операций в нем происходит за $O(1)$, после чего мы выполняем операцию вставки/поиска для $(k / 2)$ -дерева, получаем, что $T(k) = O(1) + T(k / 2)$. Очевидное решение этого уравнения $T(k) = O(\log(k)) = O(\log(\log(U)))$. Получается, что мы добились требуемой асимптотики для вставки и поиска.

К сожалению, такая структура с одними лишь операциями вставки, поиска и удаления, мягко говоря, в большинстве случаев бессмысленна, учитывая то, что в данном случае проще было создать обычный массив A , в x -ом элементе которого хранится, существует ли число x в нашем множестве или нет.

Пришло время исправлять ситуацию! Для этого реализуем, скажем, операцию $\text{FindNext}(x)$. Напомню, что эта операция ищет минимальное число q , содержащееся в нашем дереве, такое, что $q \geq x$.

Для этого немного дополним нашу структуру. Напомню, что в самом начале я сказал, что мы будем хранить в k -дереве не только $2^{k/2}$ штук $(k / 2)$ -деревьев, но минимум в нём, максимум и еще одно вспомогательное $(k / 2)$ -дерево, назовем его aux (от англ. *auxiliary* — вспомогательный). Чем же оно будет нам так помогать?

Возьмем k -дерево с его поддеревьями $\text{children}[0..2^{k/2} - 1]$ и вспомогательным деревом aux . В aux мы будем хранить все такие числа p , что дерево $\text{children}[p]$ не является пустым, т. е. содержит хотя бы одно число. Соответственно, если p -ое дерево $\text{children}[p]$ является пустым, то число p не содержится в aux .

Также, произведем незначительную модификацию: для любого дерева T мы не будем хранить минимальное число в его поддеревьях $T.\text{children}$, будем просто записывать его в переменную $T.\text{min}$. Когда в запросе вставки нам придется вставить в него число x , меньшее, чем $T.\text{min}$, то так как $T.\text{min}$ не содержится в наших поддеревьях, вставим его в поддерево, после чего присвоим $T.\text{min} = x$.

Заметим, что теперь мы не будем отдельно рассматривать случай $k = 1$, так как у нас присутствуют переменные $T.\text{min}$ и $T.\text{max}$. И если, скажем, 1-дерево содержало числа 0 и 1, то у

него просто будет $T.\text{min} = 0$, а $T.\text{max} = 1$. Если же, например, оно содержало только число 1, то у него будет $T.\text{min} = T.\text{max} = 1$.

Теперь рассмотрим алгоритм вставки в дерево с учетом всего вышеперечисленного:

```
insert(T, x):
```

```
if T.is_empty():
```

```
T.min = T.max = x
```

```
else:
```

```
if x < T.min:
```

```
    swap(x, T.min) # мы не храним
минимум в поддеревьях, придется вставить
старый минимум
```

```
if x > T.max:
```

```
T.max = x
```

```
if T.k != 1: # если это 1-дерево, то
не имеет смысла дальше что-то пропихивать
```

```
if T.children[high(x)].is_empty():
```

```
    insert(T.aux, high(x)) # если
так, то следующий insert выполнится за  $O(1)$ 
```

```
insert(T.children[high(x)], low(x))
```

Асимптотика времени его работы будет все так же $O(\log(\log(U)))$, она оценивается аналогично предыдущей версии функции insert . Необходимо лишь заметить, что если мы выполним вставку в aux , то следующий за ним insert будет работать за $O(1)$, так как это поддерево будет пустым.

Функция поиска немного поменяется:

```
find(T, x):
```

```
if T.is_empty():
```

```
return False
```

```
else if T.min == x or T.max == x: #
условие T.max == x можно было и не добавлять
```

```
return True
```

```
else:
```

```
return find(T.children[high(x)], low(x))
```

Теперь рассмотрим функцию $\text{FindNext}(x)$, вот её псевдокод с комментариями: ▶

```
find_next(T, x):
```

```
    if T.is_empty() or x > T.max:
```

```
        return None # следующего числа за x в
        этом дереве не существует
```

```
    if x <= T.min:
```

```
        return T.min # так как оно
        минимальное, оно и будет следующим
```

```
    if T.k == 1: # рассмотрим случай 1-дерева
```

```
        if T.max == x:
```

```
            return T.max
```

```
        else:
```

```
            return None
```

```
    if not T.children[high(x)].is_empty() and
    low(x) <= T.children[high(x)].max:
```

```
        # случай, когда число, которое мы
        ищем, начинается на high(x)
```

```
        return merge(high(x), find_next(T.
        children[high(x)], low(x)))
```

```
    else:
```

```
        # иначе найдем первое возможное
        начало числа, которое мы ищем
```

```
        next_high = find_next(aux, high(x) +
        1) # найдем следующее непустое поддерево
```

```
        if next_high != None:
```

```
            return merge(next_high,
            T.children[next_high].min) # дерево непусто,
            возьмем его минимум
```

```
        return None
```

```
merge(high, low):
```

```
    return результат склеивания (k /
    2)-битовых чисел high и low
```

Асимптотика её работы, очевидно, $O(\log(\log(U)))$.

Думаю, написание таких функций как Remove, FindPrevious и прочих не должно вызвать особого труда, ибо пишутся они все аналогично, поэтому их псевдокод я опущу.

Применение

Кроме очевидного применения дерева ван Эмде Боаса можно придумать множество неожиданных, например:

● **Сортировка последовательности из N чисел за $O(n * \log(\log(U)))$**

Действительно, вставим все наши числа в дерево. После чего возьмем минимальное число в дереве и будем последовательно выполнять операцию $x = \text{FindNext}(x + 1)$, в результате чего в переменной x побывают все наши числа в отсортированном порядке. Заметим, что можно написать различные реализации этой сортировки, в том числе и ту, которая заодно удаляет повторяющиеся элементы. Основное преимущество этой сортировки в том, что по асимптотике этот алгоритм обгоняет даже цифровую сортировку.

Нахождение наидлиннейшей возрастающей подпоследовательности за $O(n * \log(\log(U)))$

Некоторые наверняка слышали о такой задаче, как нахождение НВП и ее решении за $O(n * \log(n))$. Для тех, кто не слышал, могут прочитать об этом [здесь](#). Основная идея оптимизации заключается в том, что бинарный поиск в массиве можно заменить на операцию FindPrevious в дереве ван Эмде Боаса.

● **Построение выпуклой оболочки $O(n * \log(\log(U)))$**

Вот это было совершенно неожиданно, правда? Напомню, задача построения выпуклой оболочки звучит так: даны N точек на плоскости, необходимо построить их выпуклую оболочку, т. е. наименьший выпуклый многоугольник, содержащий все эти точки. Существует [алгоритм Грэхэма-Эндрю](#), который решает задачу за $O(n * \log(n))$, и который очевидным образом можно оптимизировать до $O(n * \log(\log(U)))$, как мы уже знаем. Правда, практическая ценность данного алгоритма достаточно сомнительна, но идея явно имеет право на существование.

● **Алгоритм Дейкстры за $O(E * \log(\log(U)))$**

Для тех, кто не знаком с алгоритмом Дейкстры поиска кратчайшего пути во взвешенном графе, предлагаю ознакомиться с [этой](#), а также с [этой](#) статьей. Реализация алгоритма Дейкстры с помощью кучи, как известно, работает за $O(E * \log(V))$, где V — количество вершин, а E — количество ребер. Но если теперь вместо кучи применить всем нам уже известную структуру данных, то получим асимптотику $O(E * \log(\log(U)))$, что не может не радовать.

● И еще много-много задач, количество которых ограничено лишь вашей фантазией :)

Минус всех этих алгоритмов заключается в том, что для слишком больших U дерево ван Эмде Боаса будет занимать большее количество памяти (грубая оценка — $O(U)$), чего, впрочем, можно попытаться частично избежать, создавая необходимые поддерева «лениво», то есть только тогда, когда они нам потребуются.

Вместо заключения

[Здесь](#) находится моя реализация дерева ван Эмде Боаса на C++. Она не претендует на безупречность, но свою работу должна выполнять. ■

Автор:

Николай Самойлов
nsamoylov.com

Веб-дизайн

тренды 2011

Между дизайном и разработкой существует тонкая грань, которая, с переходом в новое десятилетие, становится всё более размытой. Достаточно ли просто нарисовать красивый макет в Фотошопе? Может быть, пять лет назад этого и было достаточно. В наши дни среднестатистический пользователь интернета требует большего. Весь лоск сайта со временем надоедает, если в нем нет никакой сути. Если ваша единственная цель поразить дизайнерское сообщество вашими яркими работами долго на плаву вы не продержитесь. 2011 год характеризуется не красотой, а функциональностью. А теперь собственно тренды нового года и ближайшей декады: виртуальная реальность, постоянная связь и интерактивный дизайн.

Как остаться релевантным дизайнером в 2011? Главная

цель дизайнера не ослепить, а завлечь. Любой дизайнер может получить «охи» и «ахи», которые легко забываются. Величайший дизайнер способен создать среду, которая чарует и пленяет пользователя так, что он даже и не думает о том, чтобы искать кнопку «назад». Несколько элементов собраны воедино, чтобы создать прекрасный мир, в котором есть все: гармония цвета, интуитивный дизайн, легкодоступная информация и быстрый отклик. Кроме того, никогда нельзя недооценивать силу простоты. Конечно, это всегда было так, но в 2011 вы не только под всепрощающей эгидой десктопов и лэптов. Сейчас ваш дизайн должен справляться с нетбуками, смартфонами и планшетами. Вы готовы?

Взглянем на ТОП 11 трендов в 2011 году.

1. Больше CSS3 и HTML5

Наконец-то можно вздохнуть с облегчением! CSS3 и HTML5 были на горизонте веб-дизайна последние пару лет, но сейчас, в 2011, мы видим взрыв этих технологий. Дизайнеры начинают уходить от технологии Flash. Вы и так знаете, что она не всегда работает хорошо, новые технологии теперь доступны вашим постоянным и потенциальным посетителям. В 2011 вы отойдете от Flash и познаете всю магию HTML5. Просто взгляните на примеры (рис1):



Теперь, когда вы взглянули на примеры, поймите, что Flash и HTML не равные соперники. Есть достаточно места для обоих в 2011 году. Проблема в том, что дизайнеры в 2010 (и ранее) неправильно использовали Flash. HTML5 снимает часть той ответственности, которую мы положили на плечи Flash, однако, HTML5 еще не может заменить отдельные элементы дизайна, которые мы можем получить только с помощью Flash.

Возможно, еще более захватывающим является тот факт, что CSS3 в полной мере доступна нам в этом году. Заменят Photoshop (не-не-не, Adobe еще не ушел на покой), ведь с помощью CSS3 можно легко создать тень для текста или прозрачности картинки. Если вы еще не начали, то самое время погрузиться в изучение CSS3 и HTML5.

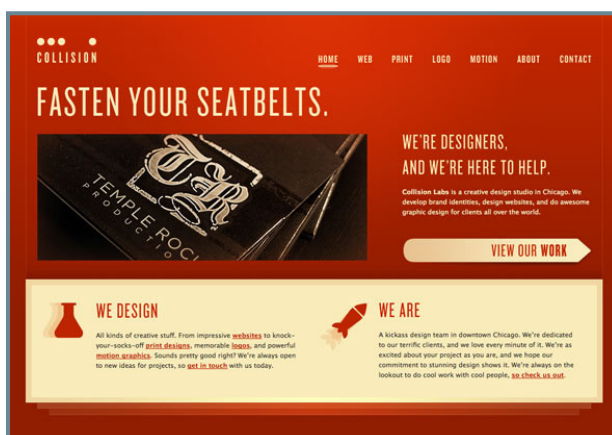
2. Простые цветовые гаммы

Простота. Нет ничего настолько же действенного как простое сообщение на спокойном фоне. Спокойствие может быть достигнуто несколькими способами. Забудьте о черном и белом, и даже об оттенках серого. Подумайте о зеленом, желтом или даже красном в качестве основных. Впрочем, ограничьте свою палитру двумя или тремя цветами. Работайте с оттенками выбранных вами цветов для разнообразия.

Ведь замечательно же, когда всего несколько цветов доносят ваше умиротворяющее послание. Смотрите:



С помощью оттенков зеленого была создана эта примочка для твиттера. Примечание: Этот сайт был создан с помощью XHTML/CSS и Javascript.



Красный может быть довольно раздражающим, если использован неправильно.

3. Mobile Ready

Смартфоны, планшеты, нетбуки, о Боже! Головокружительное количество мобильных гаджетов доступны покупателю в 2011 году. Это значит, что веб-дизайн должен быть кроссплатформенным.

Чтобы создать сайт, подготовленный для мобильных платформ не достаточно убрать «свистелки» из дизайна. Из-за этого дизайн может получиться пустым и безликим. Хотя не исключено то, что если вы уберете часть фишек из дизайна, ваш бренд не подвергнется видимым изменениям. К счастью технологии всё больше помогают нам, избавляя от этого бремени.

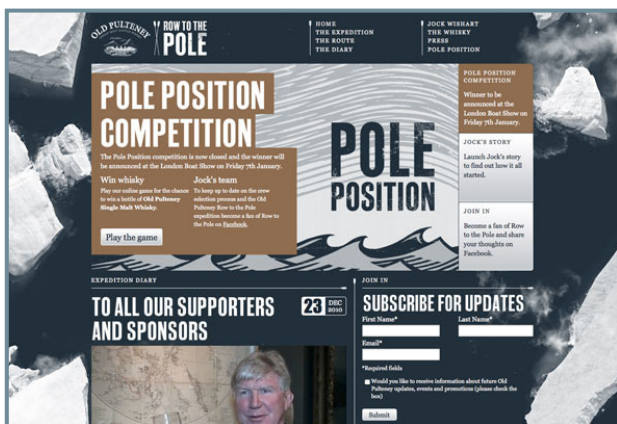
Веб-дизайн для мобильных устройств шагнул далеко вперед с помощью CSS3. Одним из наиболее важных достижений является то, что вы можете спроектировать дизайн всего сайта и с помощью средств разработки подогнать его под устройство пользователя.

Очень заманчиво создать версию сайта для мобильной платформы, однако это больше не удовлетворяет современную аудиторию. Все чаще у мобильных версий есть ▶

возможность посетить оригинальный сайт. Подготовьте ваш дизайн для удовлетворения потенциального спроса.

4. Параллакс скроллинг

Параллакс скроллинг предназначен не только для олдскульных видеоигр. Как отмечалось ранее, один из трендов на 2011 год — создание ощущения глубины. Что может быть удобнее для создания этого ощущения, чем параллакс скроллинг? Он использует слои для создания иллюзии трехмерного пространства. Эффекта можно добиться с помощью простых CSS трюков или jQuery плагина Spritely. Самый эффективный вариант использования многоуровневого скроллинга в качестве вторичного элемента дизайна. Например, шапка, футер или фон. Сделав его неотъемлемой частью навигации, вы крайне разочаруете пользователя.



5. Проектируйте для тач скрина, не для мыши

Технологии стали намного более осязаемы. Юзабилити преобразовывается из абстрактного в нечто осязаемое. Тач-скрин используется планшетами, большинством смартфонов и некоторыми десктопами. Приспособлен ли ваш дизайн к управлению пальцами?

Сколько из ваших работ ориентированы на мышь? Как дизайнеры, мы используем непосредственно мышь. Когда мы наводим на ссылки, они подсвечиваются (Примечание переводчика: «Если дизайнер не полный «чудак», то подсвечиваются»). Как ни странно у тачскрина нет наведения как такового. Как ваш дизайн покажет пользователю, что это ссылка? Что на счет выпадающих меню? Это так же не взлетит при использовании тачскрином.

Как посетителю ознакомиться с вашим сайтом? Достаточно спорный момент, однако для тачскрин-ориентированного сайта горизонтальная прокрутка может оказаться куда более удобной, чем вертикальная. Идеально вписывается в эту

нишу концепция журнала, где посетитель фактически листает странички вашего сайта.

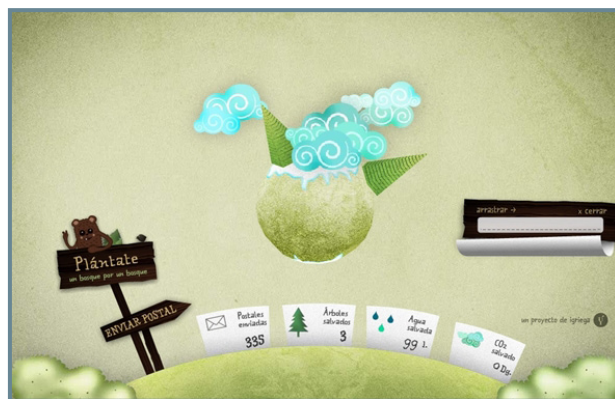
И наконец, рассмотрим возможность использования плавающего макета, как части вашего интерактивного дизайна. В 2011 вы уже не будете иметь дело с разрешением экрана. Пользователи легко могут изменить вид просмотра с вертикального на горизонтальный. Ваш дизайн должен быть гибким, чтобы удовлетворить любые прихоти, иначе он станет ископаемым.



Ребенок смотрит на iPad. Фотография Стива Пейна, Flickr

6. Восприятие глубины

Нет, я не говорю сейчас об эфемерном «Я вижу твою чашку кофе и клавиатуру». Суть глубины восприятия — объем, да такой, что некоторые элементы вашего сайта кажутся ближе, ►

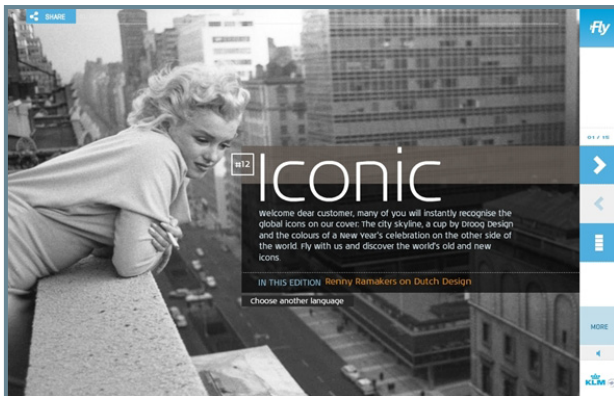


чем другие. Это создает отличный 3D-эффект, когда сделано умело. Вспомните, каково это смотреть 3D блокбастер «Аватар»? Предметы буквально выпрыгивают из экрана.

Хотя 3D технологии еще толком и не добрались до веб-дизайна, вы можете воссоздать этот объем.

7. Большие фотобэкграунды

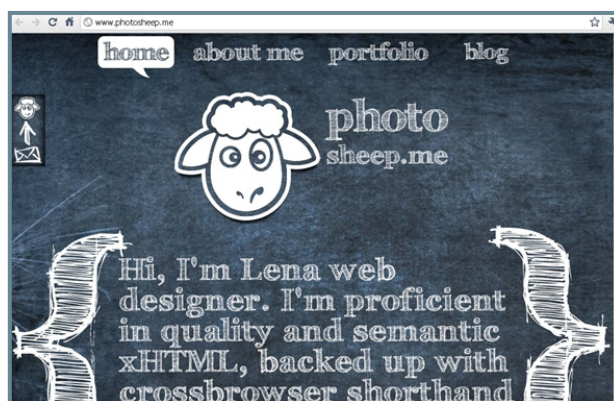
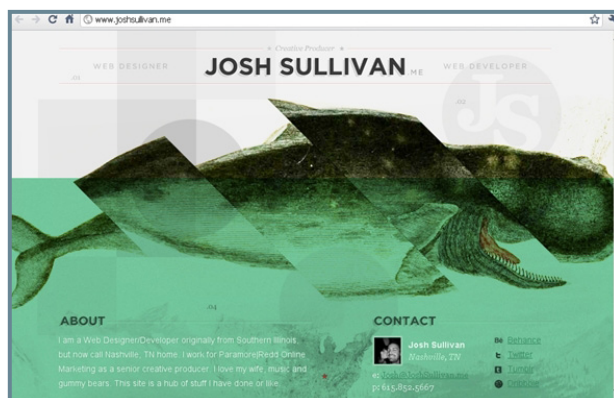
Количество больших фонов-тянучек возрастает в 2011. Эти изображения высокого разрешения и покрывают сайт целиком. Большие фото — легкий способ захватить аудиторию. Естественно фоновое фото должно быть связано с контентом. Разместив милую картинку, которая будет выпадать из контекста, вы навредите юзабилити. Тренды говорят нам о том, что лучшим выходом будут слегка прозрачные образы, не бросающие тени на контент, гармонично вписываясь в него.



8. Интересные доменные имена и их использование

Это конечно не самый острый вопрос в веб-дизайне, однако очень хочется увидеть по-больше оригинальных доменов. Когда-то желанное .com потеряло свою привлекательность, в первую очередь потому, чтобы найти свободный домен нужно придумать еще и свой язык, вроде клингонского (баловство переводчика, в оригинале — язык Na'Vi). В 2011 году

мы замечаем более редкое использование домена .com и учащенное использование таких причудливых доменов как .me и .us. Задумайтесь о возможностях и хватайте их до того, как они исчезнут.



9. QR: Quick Response



Если вы видели эти квадратные штрих-коды появившиеся на журналах, визитных карточках и еще много где, вы знаете что они — горячий тренд 2011 года. Как это перейдет в веб-дизайн? Удивительно хорошо, на самом деле.

Эти штрих-коды называются QR, коротко от Quick Response. ▶

Просто сфотографируйте уникальный штрих-код вашим телефоном, и как по волшебству откроется сайт, к которому этот штрих-код относится. Удобство QR-кодов в том, что есть миллиарды способов их использования. Разместите его на сайте для того, чтобы у посетителей была ссылка на мобильную версию. Так же вы можете отслеживать ваших посетителей через QR-коды, размещая специальный реферальный код в URL. Когда вы комментируете статьи на сайте, используйте QR-код в качестве аватара.

2011 год весьма «мобильный» и полезно использовать эту относительно новую техническую фишку.

10. Превью (миниатюры)

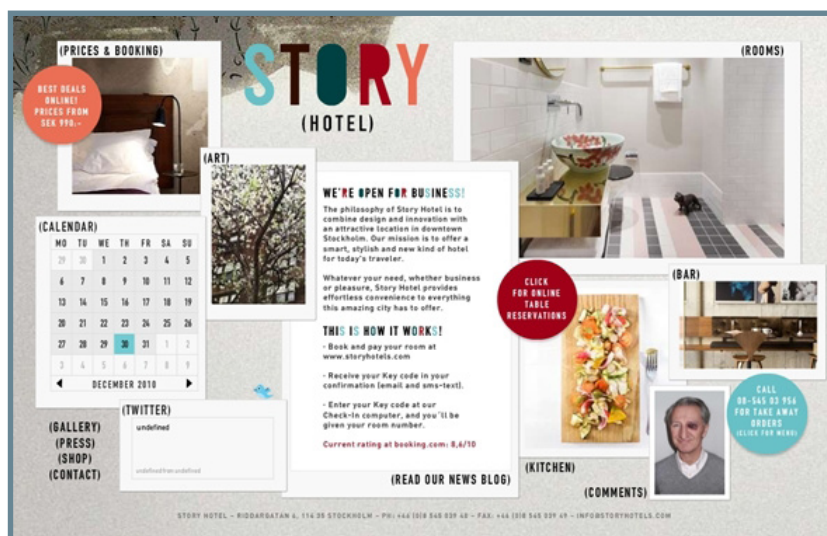
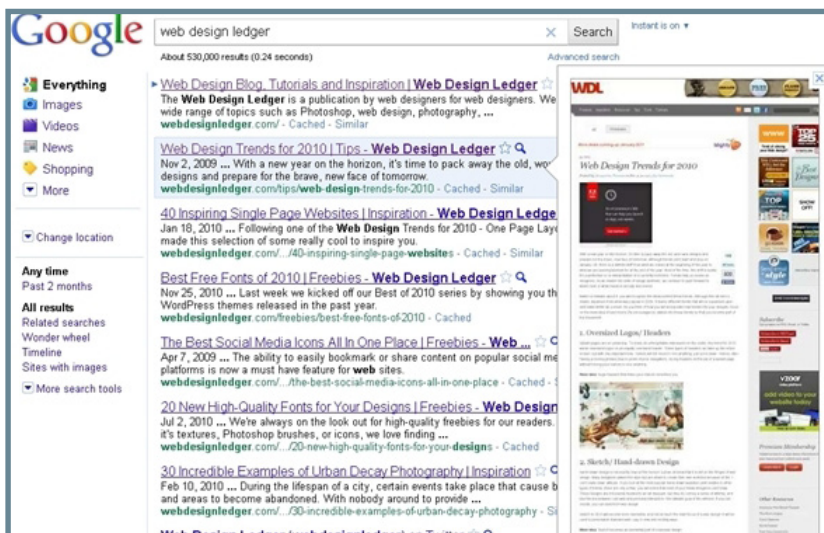
Предприимчивые люди из Google представили обычному пользователю возможность просмотра поисковой выдачи с помощью превью. Прошли те дни, когда нужно кликать по ссылке, чтобы увидеть содержимое сайта. Сейчас нужно просто нажать на «лупу» или навести на ссылку (это в том случае, если у вас не тачскрин). Перед вами магическим образом предстает то, что находится по ту сторону клика.

Если ваш сайт использует флеш, с превью возникнут проблемы. Превью не будет отображать флешевые элементы дизайна.

Так как среднестатистический пользователь становится все более и более подкованным в серфинге ожидайте, то в 2011 больше людей используют миниатюры в качестве средства навигации. Ведь так велик соблазн оценить сайт по его миниатюре.

11. Постоянная связь / Лайфстриминг

Последнее, но не менее важное, это постоянная связь. Интернет по своей сути абсолютно стерильная среда. Мы делаем его человечным, рассказывая о наших жизнях в открытой беседе, ожидая больше личных подробностей. Характерная особенность блогов



и портфолио в 2011 - это полная интеграция с Twitter'ом (это не просто ссылка на страничку в нем), а люди все чаще говорят вам, где они сейчас с помощью Foursquare. ■



CMS

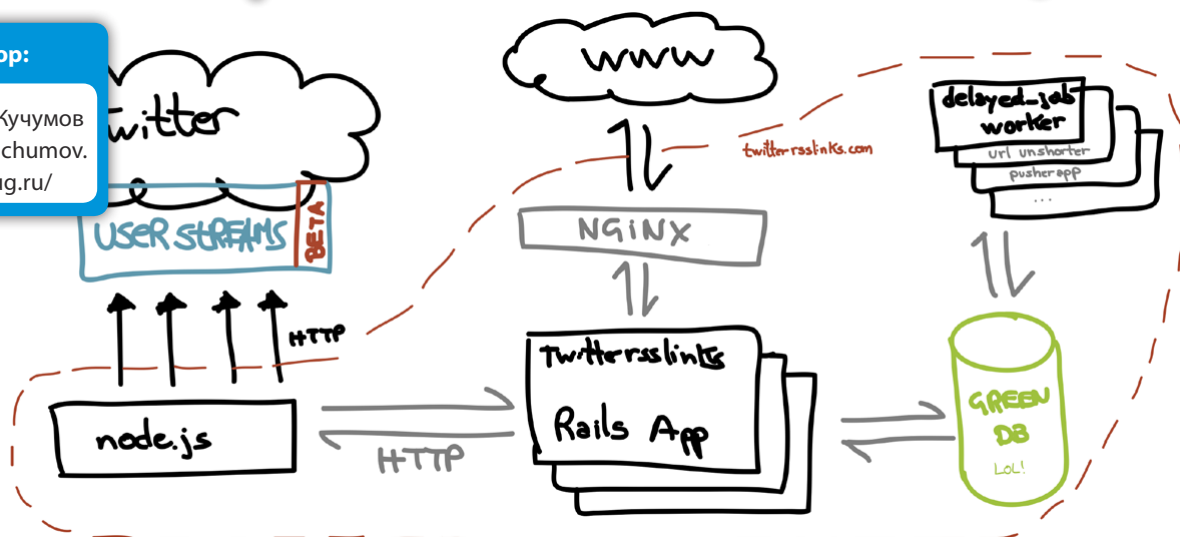
**Что такое «асинхронная
событийная модель», и почему
сейчас она «в моде»**

**Шаблоны проектов на основе
PasteScript**

Что такое «асинхронная событийная модель», и почему сейчас она «в моде»

Автор:

Николай Кучумов
nikolay-kuchumov.
moikrug.ru/



Сейчас в тематических интернет-сетях модно слово «Node.js». В этой небольшой статье мы попробуем понять («на пальцах»), откуда всё это взялось, и чем такая архитектура отличается от привычной нам архитектуры с «синхронным» и «блокирующим» вводом/выводом в коде приложения (обычный сайт на PHP + MySQL), запущенного на сервере приложений, работающем по схеме «по потоку (или процессу) на запрос» (классический Apache Web Server).

О производительности

Современные высоконагруженные сайты типа twitter'a, вконтакта и facebook'a работают на связках вида PHP + Apache + NoSQL или Ruby on Rails + Unicorn + NoSQL, и ничуть не тормозят. Во-первых, они используют NoSQL вместо SQL. Во-вторых, они распределяют запросы («балансируют») по множеству одинаковых рабочих серверов (это называется «горизонтальным масштабированием»). В-третьих, они кешируют всё, что можно: страницы целиком, куски страниц, данные в формате Json для Ajax'овых запросов, и т.п. Кешированные данные являются «ста-

тикой», и отдаются сразу серверами наподобие NginX'a, минуя само приложение.

Я лично не знаю, станет ли сайт быстрее, если его переписать с Apache + PHP на Node.js. В тематических интернет-сетях можно встретить как тех, кто считает системные потоки медленнее «асинхронной событийной модели», так и тех, кто отстаивает противоположную точку зрения.

Думая о том, на чём писать очередной проект, следует исходить из его задач, и выбирать ту архитектуру, которая хорошо накладывается на задачи проекта.

Например, если ваша программа поддерживает множество одновременных подключений, и постоянно пишет в них, и считывает из них, то в таком случае вам определённо следует посмотреть в сторону «асинхронной событийной модели» (например, в сторону Node.js'a). Node.js отлично подойдёт, если вы хотите перевести какую-нибудь подсистему на протокол WebSocket.

- Примеры систем, которым хорошо подойдёт «асинхронная событийная модель»:

- система в диспетчерской такси, следящая за перемещением каждого автомобиля, распределяющая поток пассажиров, высчитывающая оптимальные пути и т.п.

- система поддержания жизнедеятельности, постоянно собирающая данные с множества раскиданных датчиков, и управляющая химическим составом, температурой, влажностью и т.п.

- организм человека (мозг — логика управления, нервная система — канал передачи данных)

- чат

- MMORPG

Что такое «блокирующий» и «неблокирующий» вводы/выводы

Разберёмся с видами ввода/вывода на примере сетевого сокета ▶

(«socket» – дословно «место соединения»), через который пользователь интернета соединился с нашим сайтом, и загружает на него картинку для аватара. В этой статье мы будем сравнивать «асинхронную событийную модель» с «привычной» архитектурой, где весь ввод/вывод в коде приложения — «синхронный» и «блокирующий». «Привычной» — просто потому, что раньше всякими «блокировками» никто не заморачивался, и все так писали, и всем хватало. Что такое «синхронный» и «блокирующий» ввод/вывод? Это самый простой и обычный ввод/вывод, на котором пишется большая часть сайтов:

- открыть файл
- начать его считывать
- ждать, пока не считается
- файл считался
- закрыть файл
- вывести считанное содержимое на экран

В случае с нашим сокетом это будет: начать слушать сокет

- считать из него первую порцию данных картинки
- ждать, пока на него не придёт вторая порция данных картинки
- считать из него вторую порцию данных картинки
- ждать, следующую порцию данных картинки
- ...
- картинка считалась
- ставим картинку на аватар пользователю

При этом в коде нашей программы возникает «блокировка», во время которой поток простаивает, хотя мог бы заняться чем-нибудь полезным. Для решения этой задачи был придуман «синхронный» и «неблокирующий»

ввод/вывод:

- начать слушать сокет
- если на нём нет новых данных, перестать слушать сокет
- если на него уже поступила какая-нибудь порция данных картинки — считать эти данные
- перестать слушать сокет

Если эти шаги выполнять в цикле, пока не будет считана последняя порция данных картинки, то мы опять же в итоге получим всю картинку полностью. С тем лишь отличием, что в этом цикле, помимо считывания данных с сокета, мы можем делать что-нибудь ещё полезное, а не простаивать под «блокировкой». Например, можно было бы считывать данные ещё и с другого сокета. Такой цикл «неблокирующего» ввода/вывода всплывёт ещё раз ближе к середине статьи.

Существует ещё «асинхронный» ввод/вывод. В нашей статье мы не будем его рассматривать, но вообще это когда мы вешаем на сокет «функцию обратного вызова» (callback) из нашего кода, которая будет вызываться операционной системой каждый раз, когда на этот сокет будет приходиться очередная порция данных картинки. И дальше уже забываем о прослушивании этого сокета вообще, отправляясь делать другие дела. «Асинхронный» ввод/вывод, как и «синхронный», делится на «блокирующий» и «неблокирующий». Но в этой статье под словами «блокирующий» и «неблокирующий» мы будем иметь в виду именно «синхронный» ввод / вывод.

И ещё, в этой статье мы будем рассматривать только «привычную» архитектуру, где приложение запущено непосредственно на операционной системе, с её системными потоками, а не на какой-нибудь «виртуальной машине» с её «зелёными потоками». Потому что внутри «виртуальной машины» с «зелёными потоками» можно творить разные чудеса, типа превращения якобы «синхронного» ввода/вывода в «асинхронный», о чём пойдёт речь ближе к концу статьи, в разделе «Альтернативный путь».

Предпосылки

Целая лавина экспериментов с новыми архитектурами приложений была вызвана тем, что традиционная архитектура решала нужды интернета на заре его развития, и, разумеется, не была рассчитана на удовлетворение эволюционировавших нужд «веб-два-нольного» интернета, в котором всё жужжит и движется.

Проверенная годами связка PHP + MySQL + Apache хорошо справлялась с «интернетом 1.0». Сервер запускал новый поток (или процесс, что почти одно и то же с точки зрения операционной системы) на каждый запрос пользователя. Этот поток шёл в PHP, оттуда – в базу данных, чего-нибудь там выбирал, и возвращался с ответом, который отсылал пользователю по HTTP, после чего самоуничтожался.

Однако, для приложений «реального времени» её стало не хватать. Допустим, у нас есть задача «поддерживать одновременно 10 000 соединений с пользователями». Можно было бы для этого создать 10 000 потоков. Как они будут уживаться друг с другом? Их будет уживать друг с другом системный «планировщик», задачей которого является выдавать каждому потоку его долю процессорного времени, и при этом никого не обделять. Действует он так. Когда один поток немного поработал, запускается планировщик, временно останавливает этот поток, и «подготавливает площадку» для запуска следующего потока (который уже ждёт в очереди).

Такая «подготовка площадки» называется «переключением контекста», и в неё входит сохранение «контекста» приостанавливаемого потока, и восстановление контекста потока, который будет запущен следующим. В «контекст» входят регистры процессора и данные о процессе в самой операционной системе (id'шники, права доступа, ресурсы и блокировки, выделенная память и т.д.).

Как часто запускается планировщик – это решает операционная система. Например, в Linux'е по умолчанию планировщик запускается где-то раз в сотую долю секунды. Планировщик также вызывается, когда процесс «блокируется» вручную (например, функцией sleep) или в ожидании «синхронного» и «блокирующего» (то есть, самого простого и обычного) ввода/вывода (например, ►

запрос пользователя в потоке PHP ждёт, пока база данных выдаст ему отчёт по продажам за месяц).

В общем случае полагают, что «переключение контекста» между системными потоками не является таким уж дорогостоящим, и составляет порядка микросекунды.

Если потоки активно читают разные области оперативной памяти (и пишут в разные области оперативной памяти), то, при росте числа таких потоков, им станет не хватать «кеша второго уровня» (L2) процессора, составляющего порядка мегабайта. В этом случае им придётся каждый раз ожидать доставки данных по системной шине из оперативной памяти в процессор, и записи данных по системной шине из процессора в оперативную память. Такой доступ к оперативной памяти на порядки медленнее доступа к кешу процессора: для этого и был придуман этот кеш. В этих случаях, время «переключения контекста» может достигать до 50 микросекунд.

В интернете можно встретить мнение, что постоянное «переключение контекста» у большого количества одновременных потоков может существенно затормозить всю систему. Однако я не нашёл однозначных и подробных численных доказательств этой гипотезы.

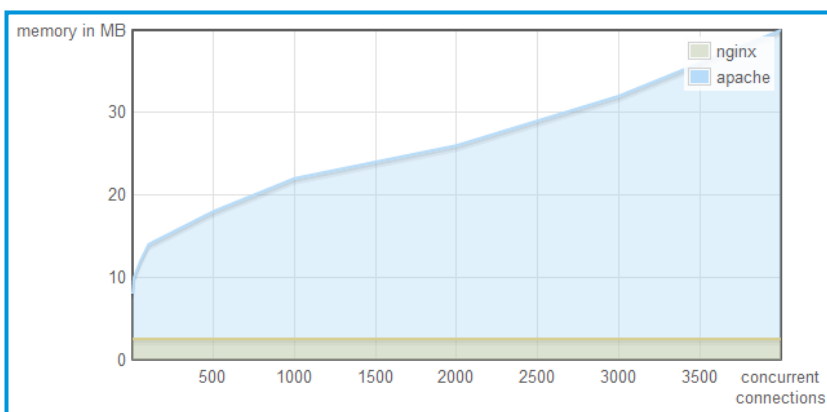
Рассмотрим ещё, какой отпечаток накладывает многопоточная модель на потребление приложением оперативной памяти. С каждым системным потоком связан «стек». Если поток вызывает некую функцию с аргументами, то в «стек» кладутся аргументы этой функции, и текущий адрес в коде, называемый «адресом возврата» (потому что по нему мы вернёмся сюда обратно, когда вызванная функция закончит выполняться). Если эта функция вызывает ещё какую-то функцию внутри себя, то соответствующие данные опять пишутся в «стек», поверх тех, которые уже были туда записаны, создавая, таким образом, подобие клубка.

При создании системного потока, «стек» выделяется операционной системой в оперативной памяти не сразу целиком, а «кусочками», по мере его использования. Это называется «виртуальной памятью». То есть, каждому потоку выделяется сразу большой кусок «виртуальной памяти» под «стек», но на деле вся эта «виртуальная память» дро-

бится на «кусочки», называемые «страницами памяти», и уже эти «страницы памяти» выделяются в «настоящей» оперативной памяти только тогда, когда в них возникает необходимость. Когда поток дотрагивается до «страницы памяти», ещё не выделенной в «настоящей» оперативной памяти (например, пытается отдать процессору команду записать туда что-нибудь), «блок управления памятью» процессора засекает это действие, и вызывает в операционной системе «исключение» «page fault», на которое она отвечает выделением данной «страницы памяти» в «настоящей» оперативной памяти.

В Linux'е размер стека по-умолчанию равен 8-ми мегабайтам, а размер «страницы памяти» — 4-рём килобайтам (под «стек» сразу выделяются одна-две «страницы памяти»). В пересчёте на 10 000 одновременно запущенных потоков мы получим требование около 80 мегабайтов «настоящей» оперативной памяти. Вроде как немного, и вроде как нет повода для беспокойства. Но размер требуемой памяти в этом случае растёт как $O(n)$, что говорит о том, что с дальнейшим ростом нагрузки могут возникнуть сложности с «масштабируемостью»: что, если завтра ваш сайт будет обслуживать уже 100 000 одновременных пользователей, и потребует поддержания 100 000 одновременных соединений? А послезавтра — 1 000 000? А после-послезавтра — ещё неизвестно сколько...

«Асинхронная событийная модель» лишена такого недостатка, и не требует новой памяти с ростом количества одновременных подключений (это называется $O(1)$). Взгляните на этот график, сравнивающий потребление оперативной памяти Apache Web Server'ом и Node.js'ом:



Современные web-серверы (включая современный Apache) построены не совсем на архитектуре «по потоку на запрос», а на более оптимизированной: имеется «пул» заранее заготовленных потоков, которые обслуживают все запросы по мере их поступления. Это можно сравнить с аттракционом, в котором имеется 10 лошадей, и 100 ездоков, которые хотят прокатиться: образуется очередь, и пока первые 10 ездоков не прокатятся «туда и обратно», следующие 10 ездоков будут стоять и ждать в очереди. В данном случае аттракцион — это сервер приложений, лошади — потоки из пула, а ездоки — пользователи сайта.

Если мы будем использовать такой «пул» системных потоков, то одновременно мы сможем обслуживать только то количество пользователей, сколько потоков у нас будет «в пуле», то есть никак не 10 000.

Описанные в этом разделе сложности, постоянно рождающие в умах вопрос о пригодности многопоточной архитектуры для обслуживания очень большого количества одновременных подключений, получили собирательное название «The C10K problem».

Асинхронная событийная модель

Нужна была новая архитектура для подобного класса приложений. И в такой ситуации, как нельзя кстати, подошла «асинхронная событийная модель». В основе её лежат «событийный цикл» и шаблон «reactor» (от слова «react» – реагировать).

«Событийный цикл» представляет собой бесконечный цикл, который ▶

опрашивает «источники событий» (дескрипторы) на предмет появления в них какого-нибудь «события». Опрос происходит с помощью библиотеки «синхронного» ввода/вывода, который, при этом будет являться «неблокирующим» (в системную функцию ввода/вывода передаётся флаг `O_NONBLOCK`).

То есть, во время очередного витка «событийного цикла», наша система проходит последовательно по всем дескрипторам и пытается считать из них «события»: если таковые имеются, то они возвращаются функцией чтения в нашу систему; если же никаких новых событий у дескриптора нет, то он не станет «блокировать» и ждать появления «события», а сразу же возвратит ответ: «новых событий нет».

«Событием» может быть приход очередной порции данных на сетевой сокет («socket» – дословно «место соединения»), или считывание новой порции данных с жёсткого диска: в общем, любой ввод/вывод. Например, когда вы загружаете картинку на хостинг, данные туда приходят кусками, каждый раз вызывая событие «новая порция данных картинки получена».

«Источником событий» в данном случае будет являться «дескриптор» (указатель на поток данных) того самого TCP-сокета, через который вы соединились с сайтом по сети.

Второй компонент новой архитектуры, как уже было сказано, – это шаблон «reactor». И, для русского человека, это совсем не тот реактор, который стоит на атомной станции. Суть этого шаблона заключается в том, что код сервера пишется не одним большим куском, который исполняется последовательно, а небольшими блоками, каждый из которых вызывается («реагирует») тогда, когда происходит связанное с ним событие. Таким образом, код представляет собой набор множества блоков, задача которых состоит в том, чтобы «реагировать» на какие-то события.

Такая новая архитектура стала «мейнстримом» после появления Node.js'a. Node.js написан на C++, и основывает свой событийный цикл на Сишной библиотеке «libev». Однако Яваскрипт здесь не является каким-то избранным языком: при наличии у языка библиотеки «неблокирующего» ввода/вывода, для него тоже можно написать подобные «фреймворки»: у

Питона есть Twisted и Tornado, у Перла – Perl Object Environment, у Руби – EventMachine (которой уже лет пять). На этих «фреймворках» можно писать свои серверы, подобные Node.js'у. Например, для Явы (на основе java.nio) написаны Netty и MINA, а для Руби (на основе EventMachine) – Goliath (который ещё и пользуется преимуществами Fibers).

Преимущества и недостатки

«Асинхронная событийная модель» хорошо подойдёт там, где много-много пользователей одновременно запрашивают какие-нибудь «легковесные» (но некешируемые) куски данных, или производят какие-нибудь «легковесные» действия. Например: получить температуру с датчика, передать изображение с видекамеры, плюсануть кого-нибудь, проставить рейтинг кому-нибудь, написать коммент, проверить, нет ли новых сообщений в почтовом ящике, отослать сообщение в чат, и т.п...

Требование «легковесности» действий проявляется, когда мы вспоминаем о том, что весь этот бесконечный цикл запущен в одном единственном потоке, и если вы вклините в этот цикл какое-нибудь тяжеловесное вычисление, то все остальные пользователи будут ждать в очереди, пока одно это вычисление не закончится.

Поэтому серверы, подобные Node.js'у, подходит только для выполнения «легковесных» задач, или как «фронтенд» для тяжеловесного «бекенда».

Ещё один возможный недостаток «асинхронной событийной модели» – иногда (не всегда, но бывает, особенно при использовании «асинхронной событийной модели» для того, для чего она не предназначена) код приложения может стать сложным для понимания из-за переплетения «обратных вызовов». Это называется проблемой «спагетти-кода», и описывается так: «коллбек на коллбеке, коллбеком погоняет». С этим пытаются бороться, и, например, для Node.js'a написана библиотека Seq.

Ещё один путь устранения «обратных вызовов» вообще — так называ-

емые continuations (coroutines). Они введены, например, в Scala, начиная с версии 2.8 (coroutines), и в Руби, начиная с версии 1.9 (Fibers). Вот пример того, как помощью Fibers в Руби можно полностью устранить коллбеки, и писать код так, как будто бы всё происходит синхронно.

Для Node.js'a была написана аналогичная библиотека node-fibers. По производительности (в искусственных тестах, не в реальном приложении) node-fibers пока работают где-то в три-четыре раза медленнее обычного стиля с «обратными вызовами». Автор библиотеки утверждает, что эта разница в производительности возникает там, где Яваскрипт стыкуется с C++'ным кодом движка V8 (на котором основан сам Node.js), и что замеры производительности нужно трактовать не как «node-fibers в три-четыре раза медленнее коллбеков», а как «по сравнению с остальными низкоуровневыми действиями в вашем коде (работа с байтовыми массивами, подключение к базе данных или к сервису в интернете), отпечаток производительности node-fibers совсем не будет заметен».

В дополнение к привычному стилю программирования, node-fibers возвращает нам ещё и привычный и удобный способ обработки ошибок try/catch'ами. Однако эта библиотека не будет внедрена в ядро Node.js'a, поскольку Райан Даль видит предназначение своего творения в том, чтобы оставаться низкоуровневым и не скрывать ничего от разработчика.

На этом основная часть этой статьи закончена, и напоследок мы вкратце рассмотрим альтернативный путь, и то, как «событийный цикл» опрашивает «источники событий» на предмет появления в них новых данных.

Альтернативный путь

В этой статье мы объяснили, почему приложение, использующее «синхронный» и «блокирующий» ввод/вывод, не выдерживает большого количества одновременных подключений. В качестве одного из решений мы предложили перевод этого приложения на «асинхронную событийную модель» (то есть, переписать приложение, скажем, на Node.js'e). ▶

Этим способом мы решим задачу фактически (закулисным) переходом с «синхронного» и «блокирующего» ввода/вывода на «синхронный» и «неблокирующий» ввод/вывод. Но это не единственное решение: мы также можем прибегнуть к «асинхронному» вводу/выводу.

А именно, мы можем использовать старый добрый «пул» системных потоков (описанный ранее в этой статье), эволюционировавший на новую ступень развития. Эта ступень развития называется «зелёные процессы» (соответственно, есть ещё и «зелёные потоки»). Это процессы, но не системные, а созданные виртуальной машиной того языка, на котором написан наш код. Виртуальная машина запускает внутри себя обычный «пул» системных потоков (скажем, по количеству ядер в процессоре), и уже на эти системные потоки отображает свои внутренние «зелёные процессы» (полностью скрывая это от разработчика).

«Зелёные процессы» — это именно «процессы», а не «потоки», так как они не имеют никаких общих переменных друг с другом, а общаются только посылкой управляющих «сообщений» друг другу. Такая модель обеспечивает защиту от разных «deadlock»-ов и избегает проблем с совместным доступом к данным, ибо всё, что имеет «зелёный процесс» — это его внутреннее состояние и «сообщение».

Каждый «объект» имеет свою очередь «сообщений» (для этого создаётся «зелёный процесс»). И любой вызов кода «объекта» — это посылка «сообщения» ему. Посылка «сообщений» от одного «объекта» другому «объекту» происходит асинхронно.

В дополнение к этому, виртуальная машина создаёт свою подсистему ввода/вывода, которая отображается на неблокирующий системный ввод/вывод (и снова разработчик ни о чём не подозревает).

И, конечно же, виртуальная машина ещё содержит свой внутренний планировщик.

В итоге, разработчик думает, что он пишет обычный код, с обычным вводом/выводом, а на деле выходит очень высокопроизводительная система. Примеры: Erlang, Actor'ы в Scala.

Как «событийный цикл» опрашивает «источники событий» на предмет появления в них новых данных

Самое простое решение, которое можно придумать — опрашивать все «дескрипторы» (открытые сетевые сокеты, считываемые или записываемые файлы, ...) на предмет наличия в них новых данных. Такой алгоритм называется «poll». Выглядит это примерно так:

- у вас есть два открытых сокета
- вы создаёте массив из двух структур, которые описывают эти сокеты
- каждому элементу этого массива вы проставляете, что и о каком сокете в него записать
- затем вы передаёте этот массив системной функции poll, которая пишет туда описание текущего состояния этих сокетов
- после этого вы проходите по этому массиву ещё раз, выясняя, есть ли там для этих сокетов новые данные
- если есть — считываете их, и делаете с ними что-нибудь
- всё это повторяется для нового витка бесконечного «событийного цикла»

Причём упомянутый массив с данными передаётся не по ссылке, а именно копируется, по причине того, что операционная система состоит из «пространства ядра» и «пользовательского пространства», которые не могут иметь общих кусков памяти (для обеспечения безопасности системы).

Поскольку доставка данных в регистры процессора из оперативной памяти, и отсылка данных из регистров процессора в оперативную память, не являются быстрыми операциями, то такое копирование массива туда-сюда сказывается на производительности системы (процессор простаивает, пока

данные по системной шине уходят в оперативную память и приходят из неё).

При этом большинство (около 95%) полученного массива (для порядка 10 000 открытых сокетов) являются бесполезными, так как соответствующие сокеты не имеют новых данных.

А раз размер этого массива растёт пропорционально количеству дескрипторов, то получается, что алгоритм этот работает тем медленнее, чем больше сокетов открыто. То есть, чем больше одновременных посетителей на вашем сайте, тем больше «событийный цикл» начинает тормозить. В таком случае говорят: «алгоритм имеет сложность $O(n)$ ».

Можно ли написать более оптимальный алгоритм? Можно, и такие были написаны в основных серверных операционных системах: epoll в Linux'e и kqueue во FreeBSD. В Windows'e также имеется IO Completion Ports, которая является своего рода близким родственником epoll'a, и была использована разработчиками Node.js'a при переносе его на Windows, для чего ими была написана библиотека libuv, предоставляющая единый интерфейс как для libev, так и для IO Completion Ports.

Рассмотрим epoll. Он отличается от простого poll'a с двух сторон.

- Программа не проверяет вообще все дескрипторы (и все виды событий), а подписывается только на те дескрипторы (и виды событий), которые ей нужны.

- Ядро делает область памяти с данными дескрипторов видимой программе путём создания файла /dev/epoll (на самом деле это «устройство», но с точки зрения философии Linux'a «всё есть файл»). Этот файл программа может читать (и писать в него) с помощью функции mmap без какого-либо копирования вообще. Наличие новых сообщений при этом проверяется системной функцией ioctl

И если обычный перебор дескрипторов занимал $O(n)$ времени, то эти оптимизированные алгоритмы требуют $O(1)$ времени, то есть не становятся медленнее с ростом количества одновременных посетителей на сайте. ■

Шаблоны проектов на основе

PasteScript



Автор:

Виталий Коробкин
www.digitaldemiurge.ru

С увеличением количества проектов встала проблема автоматизации создания каркаса под новые приложения. До недавнего времени новые проекты мы создавали путем копирования заготовленного шаблонного проекта и изменения различных настроек в разных местах проекта. Само собой это убивало достаточно много времени.

Попытки решить эту проблему вывели меня на проект под названием Python Paste. В общих чертах — это набор утилит для создания веб-приложений, к примеру можно создать свой фреймворк (в Pylons используют именно его). Помимо утилит для веба Python Paste содержит модуль Paste Script, с помощью которого можно создавать шаблоны-заготовки и на основе них генерировать проекты.

Как это все работает?

- Создаем проект, в котором будут содержаться заготовки, указываем пути до каталогов с заготовками и делаем другие настройки.

- В каталоге шаблона создаем файловую структуру, с учетом того что файлы и имена каталогов будут пропускаться через шаблонизатор CheetaH (внутри файлов можно использовать различные переменные, такие как название проекта и любые другие как сами настроите).

- Устанавливаем наш проект, содержащий заготовки.

- Командой генерации создаем новый проект на основе шаблона. Перед генерацией спросят название проекта и различные параметры, которые были указаны в настройках шаблона.

Создадим простой шаблон для проектов на Django

Устанавливаем PasteScript `pip install PasteScript`,
Создаем примерно такую структуру проекта:



Теперь определим расположение шаблонов и настроек к ним:

```
from setuptools import setup, find_packages
setup(
    name=>paster-templates>, ▶
```

```

version = "0.1",
packages = find_packages(),
install_requires=[
    <setuptools>,
    <PasteScript>,
    <Cheetah>,
],
include_package_data=True,
zip_safe=False,
# Здесь описываем наши шаблоны
entry_points="""
    [paste.paster_create_template]
    simple=templates.pastertemplates:
    SimpleTemplate
    """
)

```

где simple — название шаблона, SimpleTemplate — класс с настройками.

В классе SimpleTemplate настроим сам шаблон, укажем каталог с файлами, краткое описание и параметр «название базы», который попросим ввести пользователя:

```

from paste.script.templates import Template
from paste.script.templates import var

class SimpleTemplate(Template):
    _template_dir = 'simple'
    summary = «Simple project»
    use_cheetah = True

    vars = [
        var(<database>,
            <Database name.>,
            default=>test_db»),
    ]

```

Зайдем в templates/ и сгенерим новый джанговский проект django-admin.py startproject simple который и будет шаблоном.

Переходим к настройке файлов шаблона. Параметры, указанные при генерации можно использовать внутри файлов шаблона, предварительно добавив к названию файла _tmpl. Эти переменные имеют вид \$(property_name), названия каталогов для пропуска через шаблонизатор такие: +property_name+. Для примера кусок файла settings.py_tmpl:



```

...
# Project name - $(project)
DATABASES = {
    'default': {
        'ENGINE': 'django.db.backends.sqlite3',
        'NAME': '$(database)',
    }
}
...

```

И еще пара замечаний по шаблонным файлам:

- Если в шаблоне пишем по русски, в начало файла надо вставить #encoding UTF-8

- Если в шаблоне надо использовать символ \$ и чтобы cheetah его не принял за свой, его надо обернуть в #raw ... \$content... #end raw

Заключение

На этом все. Устанавливаем пакет с шаблоном python setup.py install.

Командой paster create --list-templates смотрим список шаблонов в системе (шаблоны ищутся, на сколько я понимаю, сканом установленных в системе egg's на предмет шаблонных плагинов, что описывали в entry_points).

Командой paster create --template=simple SimpleProject создаем новый проект на основе шаблона simple.

Осталось создать шаблоны на все случаи жизни.

Исходники проекта на GitHub. ■

Ссылки:

- django-project-templates — готовые шаблоны на основе paster'a
- ZopeSkel — набор шаблонов для проектов на Zope



Стартап

**Павел Башмаков, Stanfy Мы
создали компанию, выложив свои
лабораторные работы в Интернет**

Стартап на авиабилетах

**Стартап по-русски сегодня и
завтра**



Автор:

Артур Оруджалиев
<http://ain.ua>

Павел Башмаков, Stanfy:

**Мы создали компанию,
выложив свои лабораторные
работы в Интернет**

Сегодня у нас в гостях – Павел Башмаков, директор и сооснователь компании Stanfy, одного из наиболее заметных игроков украинского рынка разработки мобильных приложений. Среди их клиентов – «Украинская Правда», Корреспондент.net, Hotline, AlterGeo и многие другие.

Как и когда появилась Stanfy и что представляет из себя компания сегодня?

Появилась компания в июне 2005 года, когда закончив четвертый курс, мы отказались идти куда-либо работать и решили сделать свою компанию. Мы

просто придумали имя, зарегистрировали домен и сделали сайт, где выложили все свои лабораторные работы:) В то время нас было трое.

Компания сейчас принадлежит мне и текущему СТО, с которым мы начинали.

Третий основатель год назад вышел из бизнеса и уехал в Италию получать степень Ph.D. Также у нас есть опционная часть для некоторых сотрудников.

Сегодня Stanfy – это 30 человек, программисты, менеджеры проектов, ►



дизайнеры, тестировщики и сопутствующие административные сотрудники. Одновременно у нас в работе находится порядка 6-8 проектов на разной стадии. Сколько проектов делаем в год – не считал, да и эта цифра мало информативна, так как проекты все разного размера, от 1 до 12 месяцев по разработке.

Этапы были разные, от переориентации с веба на мобайл разработку, до организации внутренней структуры компании, когда начали появляться отделы и руководители в них.

Еще можно сказать, что мы в среднем меняли офис раз в два года, но из последнего нашего офиса мы успели вырасти за год и сейчас нам уже тесновато; и в ближайшее время у нас должен появиться второй офис.

Когда пришло понимание, что “лабораторные работы” представляют уже из себя бизнес?

Лабораторные работы в бизнес превратились наверное через год, когда мы решили брать первого нашего сотрудника и еще месяца через 3 искать наш первый офис. Первых клиентов привлекали на биржах труда и сайтах по поиску фрилансеров, также тогда же сработал канал по друзьям.

Почему вы специализируетесь на разработке и аутсорсинге, а не выпуске своих продуктов? Был ли такой опыт?

Мы начали с разработки для клиентов и под это в компании выстраиваются процессы. Продуктовая же модель действительно иногда бывает лучше, но для этого должны быть немного

другие люди и другие процессы. Сейчас у нас два больших направления – сильная разработка софта под заказ, и платформенные решения, с помощью которых мы сможем делать быстро мобильные приложения для наших клиентов.

В лаборатории у нас есть некоторые начинания, которые смогут в будущем претендовать на роль нашего продукта, но об этом пока рано говорить. У нас был опыт разработки своего продукта, например, игрушка для iPhone Little Journey, которую мы два месяца создавали и на которой заработали ровно ноль. А так серьезно готовились к запуску, думали «Ща как жажнет!» и не жажнуло. Так как надо больше внимания и времени уделять продвижению, постоянной работе с пользователями приложения и т.д.

Основное отличие от разработки на заказ – это жизнь с продуктом и все время отслеживание его показателей, а на это не хватало времени и, наверное, желания. В последствии у нас купили права на использование нашей игрушки, но это уже другая история, и денег она не принесла много.

Часто ли работаете на иностранных заказчиков? Как находите клиентов за границей? А в Украине какой канал привлечения клиентов основной?

С иностранными заказчиками работаем часто, сейчас это, может быть, половина загрузки всей нашей команды и планируем увеличивать количество таких клиентов. На счет основного канала продаж – сейчас это 100% пассивные продажи. Наиболее эффективны повторные обращения от наших же клиентов и построение долгосрочных отношений с ними.

Второй по значимости канал – это рекомендации друзей и довольных клиентов. Цены никак не аргументируем, есть часовые ставки специалистов и процесс работы над проектом, в результате которого получается качественный продукт. И из нашего опыта мы знаем сколько надо потратить время на создание того или иного проекта в запрашиваемом качестве. Этому не понимают неопытные заказчики, которые не имеют дело с софтом и которым важно количество фич, а не реальное



качество получаемого продукта.

Мы несколько раз пробовали активные продажи, но ни разу не добились в этом успеха. Сейчас весь наш рост происходит за счет нашего опыта и репутации на рынке.

Какие свои приложения считаешь наиболее успешными и почему?

Из того, что принадлежит Stanfy, мне нравится UA Today для Android, потому что мне нравится как он сделан и сколько людей им уже пользуется. Из того, что мы сделали для клиентов, мне нравится AlterGeo и еще один проект, который в течении этого месяца должен будет уйти в AppStore, называть его пока не буду. Hotline тоже нравится.

Я все проекты люблю, так как я являюсь неким барьером качества на проектах, и иногда по моей вине мы доделываем проекты себе в минус, чтобы они стали лучше и удобней для пользователя.

Сколько, по твоим оценкам, в Украине подобных разработчиков (именно компаний)? Чувствуется ли конкуренция? Как рынок меняется со временем?

Конкуренция чувствуется, когда клиенты нам сообщают, что мы тут еще с несколькими компаниями вместе участвуем в тендере или когда нам сообщают, что наши цены в разы выше чем у наших конкурентов. И это нормально, все больше и больше компаний выходят на рынок мобильной разработки, так как видят тренд, который создаем мы и наши клиенты.

В Украине около 5-7 компаний еще можно найти в Гугле, которые занима-





ются разработкой под iPhone / Android и у которых что-то есть в портфолио. Изменение рынка замечаю в том, когда дешевая цена разработки привлекает клиентов, которые обжигаются не качественной работой и начинают искать системных подрядчиков с хорошим опытом. Очень сильно этот рынок сдерживает отсутствие официального App-Стор в Украине.

Создается впечатление, что у вас больше продуктов под iOS, чем под Android? Так ли это? Это скорее продиктовано рынком или вашим курсом?

Клиенты больше интересуются iOS. Даже когда они приходят с двумя платформами на уме, всегда начинают с iPhone и только потом подтягивают Android. Но за последнее время наша команда Android растет быстрее, чем iPhone; и со временем тренд уйдет в Android или по крайней мере сравняется.

Какова стоимость разработки приложения сегодня на украинском рынке (было

бы здорово узнать под разные продукты и разные платформы, примерные цифры)?

За весь рынок сложно сказать, так как это все – «средняя температура по больнице». Зная, что за \$2-3K вы можете себе заказать разработку мобильного приложения, даже у нас за такие деньги есть для клиентов предложения на нашей платформе Mobile News. Если же что-то сложнее и для чего нет готового решения на базе существующей

платформы, то цена идет к \$5K и выше.

Но всегда есть много способов удешевления разработки, от подождать пока появятся готовые инструменты для вашей задачи, до перенести все на HTML5 и самому сделать мобильное приложение. Все зависит от ваших целей, аудитории и бюджета. Иногда оптимизированные HTML страницы с помощью CSS смогут вам лучше решить задачу, чем приложение, в каждом случае надо смотреть индивидуально.

P.S. Интервью с представителем ВКонтакте задерживается по техническим причинам. Ответы на вопросы из комментариев (обновлено):

Что думаете о средах кросс-платформенной разработке под мобильные платформы (например Titanium и другие). Есть в них будущее?

Думаю, что есть. Сейчас их уже можно активно использовать в корпоративном сегменте, когда красоты во втором плане, и важен только функционал. Мы плотно смотрели на Titanium

в частности, и нам он пока не подошел; но будущее у них точно есть.

Делаете сейчас что-то для Android Honeycomb?

Пока нет, все только смотрят и ждут когда же девайсов станет больше, и мы такие же 😊

Готовы ли вы взять на работу человека с минимальными знаниями в области мобильной разработки или вообще без них, но с большим желанием заниматься этим?

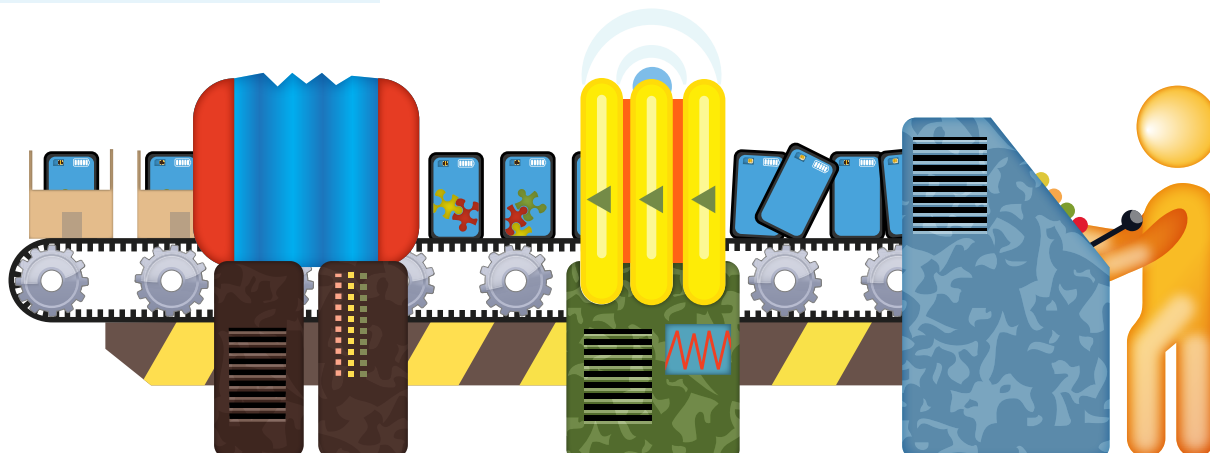
Мы сейчас активно растем, и в этом вопросе главное не лопнуть. Присылайте резюме на одну из наших вакансий, там посмотрим.

Как вы считаете есть ли потенциал роста количества пользователей iPhone в Украине? Даст ли какой-то существенный прирост возможный старт официальных продаж?

Думаю прирост будет, но не очень большой. А вот прирост активности в разработке приложений точно будет с открытием локального App Store.

Что дало бы его присутствие? Что изменилось бы? Способы оплаты? Почему им нельзя пользоваться сейчас?

Способы оплаты не основное, а вот в головах у обычных людей появилась бы галочка, что можно просто купить приложение за 9 грн. с карточки любимого банка. А когда карточки не всех банков принимаются и надо регистрировать аккаунт то в России, то в Штатах – это не масс-маркет. ■



Стартап на

авиабилетах

**Автор:**Петр Кутис
www.forbes.ru

Петр Кутис, один из создателей Apuwayanpuday.ru, объясняет, как намерен покорить рынок онлайн-продаж авиабилетов со своим новым проектом

Онлайн-продажи билетов — популярный бизнес. Наберите в «Яндексе» «заказать авиабилеты по интернету», и поисковик выдаст вам десятки и сотни сайтов, предлагающих эту услугу. По сути бизнес выглядит несложно: стань членом IATA (Международная ассоциация воздушного транспорта), создай агентство или договорись с существующим, подключись к глобальной системе дистрибуции авиабилетов (Galileo, Amadeus и т.д.), напиши софт, который поможет клиенту выбрать оптимальный вариант, прими оплату кредиткой, отправь клиенту электронные билеты и получи свою комиссию. Впрочем, простота эта обманчива. Чем привлечь клиента именно к себе? Уникальным товаром? Но сотни конкурентов предлагают, по сути, одни и те же рейсы. Дешевизной? Но ты же посредник, и не всегда сможешь продать билет дешевле, чем сама авиакомпания. В 2008 году Петр Кутис вместе с партнерами запустил сервис Apuwayanpuday.ru. Однако в прошлом году, решив, что сможет сделать все еще лучше, создал с нуля собственный билетный сайт — OneTwoTrip.com. В чем фишка?



Бизнес-модель

Мы запустились полноценно три месяца назад — в мае. Пока у сайта около 30 000 уникальных посетителей в месяц, они покупают в среднем 100 билетов в день. Средняя стоимость билета — \$400, наша доходность — 3-4%, затраты — около \$50 000 в месяц. На окупаемость планируем выйти уже в сентябре, а в конце года количество продаваемых билетов в день будет достигать нескольких сотен.

Мы зарабатываем на комиссии, которую нам перечисляют агенты. Те же за каждый проданный билет получают комиссию и с авиакомпании, и с глобальных систем дистрибуции авиабилетов (GDS). Весь процесс полностью автоматизирован, агент не тратит ничего: он дает виртуальный пульт, который позволяет от его имени выписывать билеты большинства авиакомпаний. Мы ввели понятие рейтинга рейса с десятибалльной шкалой и пятью звездами. Он включает информацию о задержках и отменах, возрасте летящего самолета, состоянии между спинками кресел. До нас в мире не было публичного ресурса, который бы консолидировал всю эту информацию. И только часто летающие пассажиры знают, где достать, например, статистику задержек, опозданий и отмен. Особенно это важно на пересадочных рейсах с короткими стыковками.

Если вы не по своей вине опоздали на стыковочный рейс, то важно знать, какая именно авиакомпания совершает перелет, а к какой предъявлять претензии. Дело в том, что в каждом рейсе есть три типа компаний: валидирующая (на ее бланке будет выписан билет), маркетинговая и непосредственно выполняющая рейс. Все три не всегда объединены одной авиакомпанией. Наш сервис сразу указывает, какая компания летит, какая продает билет и кому, следовательно, предъявлять претензии. При этом мы даем рейтинг именно по каждому конкретному рейсу (направление, время вылета и география полета). В этом рейтинге учитывается квалификация пилотов, качество обслуживания в аэропортах, уровень работы менеджеров. Ведь нужно иметь в виду, что иногда невылет рейсов связан с тем, что руководство авиакомпании не умеет договариваться.

У нас есть бесплатные источники информации, а есть платные. Фактически рейтинг строится на инсайде. Например, откуда знаем возраст борта, который может лететь? Мы собираем данные у сообщества радиолюбителей, которые консолидируют эту информацию, передаваемую пилотами авиадиспетчерам в открытом виде, определяем частоту тех или иных бортов, выполняющих различные рейсы, и с большой долей вероятности знаем, какой именно борт совершит тот или иной рейс. Еще существует общая статистика о том, какой самолет у какой авиакомпании и когда приобретен, когда выпущен, когда впервые слетал.

Мы говорим, какой багаж и какое ограничение по багажу употребляется на этом участке. Показываем сетку полетов, в ней по-разному раскрашены даты, в которые

есть прямые рейсы и рейсы с пересадкой. Еще проверяем, какие авиакомпании с какими стыкуются, а какие нет, чтобы получить билеты одного перевозчика. Показываем на этапе выбора, какие билеты возвратные, а какие нет. Если человек никогда в декабре в Майями не летал, мы покажем, какая там погода. Если человеку нужно в город, в котором нет аэропорта, сразу на карте покажем, где ближайшие аэропорты и как туда проехать.

В зависимости от наличия прямых и пересадочных рейсов клиенту показывают разные варианты: самый быстрый, самый надежный или самый дешевый рейс. Для всего этого мы покупаем различные базы данных, которые обновляются с периодичностью от нескольких раз в день и все то стоит немалых денег.

На чем экономит покупатель

При оплате билетов, в 99% случаев деньги покупателя перечисляются напрямую в авиакомпанию, то есть мы не пропускаем деньги через себя. Во-первых, это резко повышает лояльность и снижает риск невылета клиента по причине неперечисления денег от продавца/агента в авиакомпанию. Во-вторых, не увеличивает конечную стоимость билета, ведь за снятие денег, так называемый эквайринг, нужно платить. В Европе он стоит минимум 1,6% от стоимости билета, а в России от 2-3%. Разумеется, эти траты обычно перекладываются на клиента. Другие сервисы не позволяют передавать карту клиента напрямую в авиакомпанию из-за технических сложностей. Например, любую карточку отдавать авиакомпании нельзя. Для начала нужно определить, не мошенники ли это. Наш партнерский банк в связке с новым платежным шлюзом позволяет нам применять такие методы защиты и проверки, которых на рынке ни у кого нет.

Сервису по продаже онлайн-билетов надо работать со многими агентами по всему миру, поскольку стоимость одного и того же билета может быть разной, в зависимости от места его выписки (курсы валют разные, разные тарифные сетки авиакомпаний, разные условия по комиссиям). Билет-то виртуальный, и выписать его можно в любой точке мира. Наш сайт по каждому поиску пользователя определяет, где билет дешевле, и предоставляет уже проверенную и отфильтрованную информацию.

Также ради экономии мы продаем билеты только по карточкам MasterCard или Visa и не работаем с кошельками вроде Webmoney или «Яндекс.деньги», которые берут дополнительную комиссию с продавца. При этом ►

средства с карты снимаются в той валюте, в которой их продает сама авиакомпания агенту (так, билеты «Аэрофлота» немецкому агенту продаются в евро). Наш сайт показывает цены относительно официального курса ЦБ, так что иногда бывает, что фактически билет обходится еще дешевле. Если человек знает, как работает вся эта система, он может сам ходить и искать онлайн-агентства в разных странах или лазить по сайтам авиакомпаний.

Лоукосты, например, EasyJet, экономят на всем, не выплачивают комиссии агентам и поэтому зачастую продают билеты только на своих сайтах. В последнее время появилась тенденция — лоукосты стали предоставлять свои тарифы в GDS, но при этом стоимость их выше, чем на сайте самой авиакомпании.

Перспективы рынка

Рынок онлайн-продаж авиабилетов в России пока на низком старте. Люди все больше летают на самолетах, при этом борта заполнены всего на 75%. Доверие к оплате продуктов и услуг банковскими карточками тоже растет. Пассажиры постепенно переходят из офлайна в онлайн (хотя, по нашим оценкам, в России в офлайне все еще продается около 90% билетов). В 2010 году, по данным крупнейших банков-эквайеров РФ, а также международных платежных систем Visa и MasterCard, в России с банковскими картами было совершено покупок электронных авиабилетов на сумму \$1,8 млрд. Это на 80% больше показателя 2009 года. Растут продажи как на сайтах авиакомпаний, так и на сайтах агентств. Но на сайтах агентств все еще покупают ничтожно малую долю билетов. Так, топ-5 авиакомпаний России реализует через онлайн агентства только около 7,9% билетов.

С 2008 года на рынке онлайн-продаж авиабилетов ничего нового сделано не было. Появляются сайты с новыми интерфейсами — но не более. Между тем появилось много новых технологий, стало доступным большое количество информации относительно авиaperелетов, которые на OneTwoTrip.com мы уже используем и продолжаем дальше придерживаться существующих трендов и зарождавать новые. Мы вводим так называемую «авиабилетную корзину»: не спрашиваем ни о количестве пассажиров, ни об их возрасте, иначе блокируется доступ к более дешевым тарифам. Клиент нашел рейс, выбрал билеты. После этого, если нужно, например, четыре билета, система сама подберет два места по минимальной цене и еще два по самой низкой из оставшихся. Сайт будет сам искать билеты не только «туда-обратно», но и отдельно в одну сторону и отдельно в другую. Так, Air Berlin зачастую дешевле купить именно по отдельности. Также будем предлагать

полететь разными авиакомпаниями, не входящими в один альянс.

Идеи для развития

Сейчас мы в процессе разработки системы продажи билетов в кредит в режиме онлайн и онлайн-регистрацию на рейсы. Кроме того, OneTwoTrip.com в течении двух месяцев запустит кобрендинговую карту и действительно прозрачную бонусную схему. За каждую покупку будут начисляться от 2% до 3% от стоимости билета. Это экономия на следующие билеты, а при покупке билетов третьим лицам бонус можно будет оставить себе, чтобы потом потратить на свой отпуск. В скором времени мы будем давать оценку качества еды, причем сами посетители будут присылать информацию после полета.

Переводим сервис на английский язык, чтобы конкурировать с такими гигантами, как Expedia.com и Priceline.com. Цены будем варьировать в том числе и в зависимости от нормы прибыли и цен других участников рынка. Будем предлагать отели, ж/д билеты и аренду машин. Сейчас входим в социальные сети, чтобы кроме прочего предлагать людям лететь вместе. Не стоит забывать, что карта на сайте может быть рекламной площадкой. Например, в каком городе в этот день выступают такие-то группы.

И, наконец, мы расскажем каждому клиенту, за что он платит, покупая билет. Обычно, если билет стоит 5000 рублей, 70% — это тариф, а 30% — это таксы. Многие не знают, что при приобретении невозвратного билета, на самом деле можно зачастую вернуть до 20% стоимости билета, возвращая топливные и страховые сборы. Многие агенты на этом зарабатывают, умалчивая про такие вещи, мы же хотим быть открытыми и честными. Раскроем, сколько каждый платит в аэропортах за прохождение паспортного контроля, таможенного контроля, сколько за проверку кошельков и овощей, хотя вы ничего такого не везете. ■



Автор:

Ольга Гончарова
ig-mag.ru

Стартап по-русски: сегодня и завтра

Тема статьи — «Стартап по-русски». Мы не могли её не затронуть, ведь о стартапах сейчас не говорит только ленивый.

Мы попытались взглянуть на формирующийся отечественный рынок стартапов с разных сторон, и думаем, что справились с поставленной задачей. К сожалению, некоторые задумки не были воплощены (так, по объективным причинам, не удалось побеседовать с представителями «Сколково»); надемся, это не помешает читателю сформировать своё представление о ситуации со стартапами в России.

Известнейшие деятели стартап-тусовки выразили своё мнение о пробле-

мах рынка, поделились планами своих проектов, дали свои советы начинающим российским предпринимателям.

Главный стартап-инвестор России Аркадий Морейнис («Главстарт»), совместно с [Ольгой Гриневской](#), рассказывает о [StartUpWeekend](#), проблемах начинающих российских предпринимателей, и о том, как привлекаются инвесторы:

...Главная проблема рынка сейчас — недостаток людей, готовых работать. Потому что при наличии команды хороших специалистов даже нежизнеспособную идею можно как-то докрутить в процессе,

и проект не умрет. А если нет команды — ни одна даже самая гениальная идея сама себя не реализует. ...Вообще, в России очень забавная ситуация: у нас скорее не проекты ищут инвестора, а инвесторы ищут проекты. Наши мероприятия и собирают столько инвесторов по очень простой причине: у нас есть вменяемые проекты.

Харизматичные Алёна Попова и Маруся Подлеснова (а для Маруси, как оказалось, это было первое официальное интервью!) — основатель и руководитель «[Стартап Афиши](#)» соответственно — поведали нам о том, как они ►

встретились, почему решили заняться проектом, и о том, что ждет участников «Стартап Афиши» в ближайшем будущем:

...Сегодня, кроме привлечения инвестиций в стартап-проекты и поиска достойных идей для вложения, я бы назвала важными проблемами поиск людей в команду проекта, нехватку аналитики по рынку в целом, жилищный вопрос, оценку качества проводимых мероприятий и получение профильного образования. У нас активно идет работа над новыми сервисами, запуск которых мы объявим в сентябре. Это будет бомба, поверьте!

Мария Кулинцева, аналитик [StartupPoint](#), ответила на наши вопросы о крупнейшем стартап-сообществе Рунета, рассказала о формате работы проекта и о том, как изменялся StartupPoint во времени:

...Мы подстраиваемся под развитие рынка. У нас расширяется база, появляется больше и больше новых партнеров. Пробуем новые форматы работы, меняется сайт (появляются новые возможности), меняются Поинты. Скоро (думаю, в сентябре) представим новую концепцию нашей работы со стартапами (пока говорить о ней немного рано) — это будет показательно.

Истинные эксперты отрасли (Валентин Домбровский, Сергей Митрофанов, Евгений Калинин, Дмитрий Калаев, Алексей Крол), помимо своих комментариев, дали ответы на актуальные для стартаперов вопросы о развитии идеи, выборе инвесторов.

Евгений Калинин (консультант по ИТ-бизнесу, тренер, ментор):

Если вы заранее сами знаете, что за бизнес хотите сделать, и довольствуетесь этим знанием — у вас проблема. Почти на каждом стартап-шоу бывает какой-нибудь проект, после выступления которого ведущий спрашивает зал: «Кто-нибудь понял, о чем этот стартап?» И автор проекта одиноко поднимает руку.

Дмитрий Калаев (советник министра

экономики Свердловской области, один из инициаторов проекта «Уральский ИТ-кластер»):

Инвестор должен быть вам духовно и эмоционально близок — это человек, с которым придется регулярно общаться в течение трех-пяти лет. Не зря иногда инвестиции сравнивают с женитьбой.

Рынок стартапов сегодня

Аркадий Морейнис:

Главные особенности русских стартаперов — другой подход к работе, меньший процент технических фаундеров и почти полная юридическая безграмотность. Как ни странно, сами идеи стартапов, что в России, что в США — почти одинаковые. Но вот работают люди по-разному.

Мария Кулинцева:

Рынок сейчас находится на стадии становления (всё ещё). Постоянно появляются новые игроки, кто-то пропадает. Гораздо больше людей стали проявлять интерес к этому бизнесу, приходить в него (по последней тенденции — все более «возрастные» люди). Крупные компании начали присматриваться к стартапам (поняли, что иногда дешевле купить проект, чем разработать технологию собственными силами).

Сергей Митрофанов, руководитель и партнер российского подразделения международной консалтинговой компании BRANDFLIGHT, эксперт в области построения брендов:

Самая же большая проблема стартапов в России в том, что стартаперы не видят свой проект как бизнес. Они видят его как интернет-проект, какую-то игрушку, «фичу», но бизнес — это зарабатывание денег. Стартаперы излишне фокусируются на инвесторов, забывая про рынок, а это приводит к усложнению

отношений между ними. Приходя к инвестору, они предлагают идею, а не участие в бизнесе.

Только тусовка или нечто большее?

Мария Кулинцева:

Многое сейчас перенимается с Запада, еще есть возможность просто скопировать какой-нибудь западный сервис и «взлететь». Мы сейчас готовим отчет о нашем исследовании инвестиций в стартапы Рунета. Когда данные будут известны, можно будет о чем-то говорить, пока рано делать анализ. Рынок еще растёт, и будущее видится «светлым и радостным»: многие инвесторы обращаются к этой области, потому что понимают, что за ней будущее.

Алёна Попова:

«Какими будут стартапы в будущем? Я верю в Smart Grid и облачные технологии».

Маруся Подлеснова:

А я голосую всеми руками и ногами за приложения для мобильных телефонов, ибо я верю в то, что мы скоро все уйдем из веба как такового в приложения. Не важно, будет это iOS или Android, важно то, что все наши потребности будут закрываться различными приложениями в наших с вами телефонах.

Сергей Митрофанов:

Будущее у российского рынка стартапов, безусловно, есть. Это видно по тому, насколько хорошие, интеллектуальные идеи наших ребят востребованы в западных корпорациях. Интеллект — это то, что будет двигать наши стартапы. Когда в России построится оптимальная инфраструктура: сейчас это проблема для стартапов, не столько IT-проектов, но и венчурных, в сфере науки и техники, испытывающей особые трудности при создании опытных образцов своей продукции. ■



ИНТЕРНЕТ

**Пользователей раздражает
регистрация на сайтах**

Завел Facebook - забудь о privacy

Маркетинг Стива Джобса

**Александр Люстик - советы и SEO-
откровения**

**С играми так - если создал хит, он
тебя кормит много лет**

Как запомнить все пароли

Стереонизображение - это просто

Пользователей раздражает регистрация на сайтах

Автор

Артем Овечкин
matik.ru

Более 80% опрошенных агентством Matik считают, что успешно регистрироваться на сайтах, форумах и интернет-магазинах им мешают пароль и капча. Таковы результаты пусть небольшого, камерного, но показательного опроса 68 компаний, который мы провели в прошлом месяце. Им был задан только один вопрос: «Что именно раздражает вас больше всего, когда вы заполняете форму регистрации на сайтах?» Большая часть респондентов отметили 2 помехи: слишком сложную капчу и ошибки при введении нового пароля.

Никто не высказался против регистраций как таковых, к ним уже привыкли. Камень преткновения в поправлении законов юзабилити в процессе создания нового аккаунта.

У юзабилити много законов, но при регистрации новых аккаунтов чаще всего нарушаются два из них.



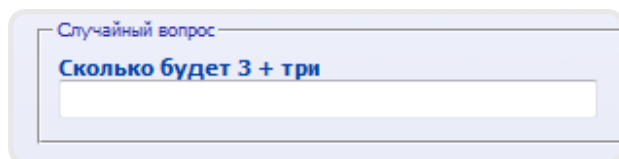
Закон №1. Капча создается против ботов, но для людей

Капча (цифро-буквенный набор, предназначенный для защиты от спамерских регистраций) не должна представлять собой головоломку, которую можно решить только путем «научного тыка». Метод проб и ошибок

здесь недопустим.

Особую сложность представляют собой «открытые» капчи в виде вопросов, потому что они допускают разные варианты написания ответа. Скажем, в данном при-

мере непонятно, надо ли вводить сумму буквами или цифрой.

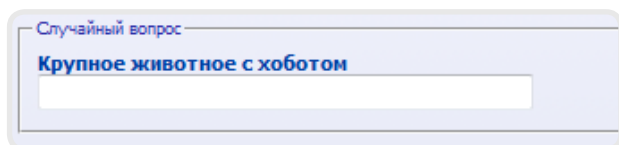


Неправильный способ написания неизбежно выдаст ошибку «неправильный ответ на вопрос», а регистрирующий должен сам догадаться, в чем она состоит.

Ошибок и связанных с ней неудобств можно избежать, приписав к вопросу подсказку: «Напишите число» или «Напишите слово».

Кто-то скажет, что такие мелочи не стоят серьезного внимания, но в юзабилити мелочей нет. Представьте, как утомляет порой регулярно наталкиваться на предупреждения о неправильно заполненной форме или ошибочно введенной капче? Как показывает опрос, нечитаемая капча останавливает около 4/5 посетителей сайтов. После 2-х неудачных попыток ввести ее почти 95% отказываются от дальнейших попыток зарегистрироваться.

Приведем еще один пример случайного вопроса с разными вариантами ответа.



Казалось бы, ответ очевиден. Однако сайт не примет ответ «Слон». И попробуй тут догадаться, в чем состоит ошибка! После 2-3х неудачных вариантов, был найден правильный ответ – «слон». Оказывается, ответ надо писать со строчной буквы. Вот они издержки хорошего образования и «дурная» привычка начинать текст с заглавной буквы. С другой стороны, что мешало автору сайта настроить алгоритм таким образом, чтобы принимались к ответу обе варианта: «Слон» и «слон». Если не хватает нужных знаний для модификации алгоритма капчи, то можно хотя бы приписать подсказку: «слово со строчной буквы». Хотя последний вариант не слишком технологичен и говорит о любительском уровне сайта.

Простой, казалось бы, вопрос: «Какой фрукт изображен на картинке?»



Думаете, яблоко? Оказывается, нет: такой ответ капча не приняла. Может, стоило написать “apple” или “плод

познания добра и зла”?

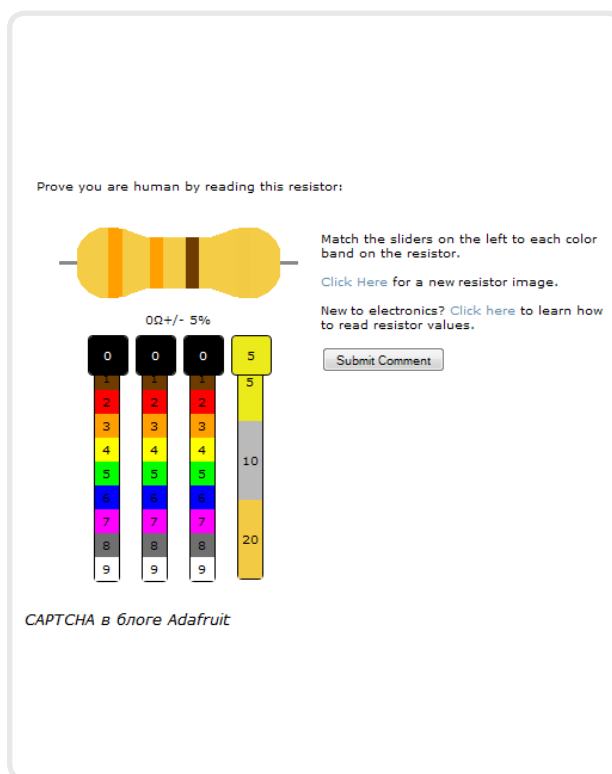
Вторая попытка. Что изображено на картинке?



Пламя? Огонь? Костер?

Ощущение, что создатели капчи очень любят телегадку «Своя игра», где нужно выбрать самый популярный ответ.

В сети встречаются и вовсе экзотические капчи, как приведенный ниже цветовой тест.



Я не буду распространяться, отстаивая интересы пользователей, страдающих дальтонизмом. Подобные извращения безопасности будут раздражать не только их, но и всех посетителей сайта. Не слишком ли высокий порог вхождения устанавливается на некоторых сайтах? Почему постоянно нужно решать какие-то задачки, отвечать на вопросы и напрягать глаза?

Зацикленность на безопасности и равнодушие к основам юзабилити приведут к тому, что пользователи станут очень избирательно подходить к регистрации на сайтах. Только самые популярные и уникальные из них смогут претендовать на привлечение зарегистрированных участников.

Как один из вариантов решения проблемы можно ►

упомануть возможность регистрации на сайте с помощью аккаунта Facebook и Вконтакте. Однако он подходит для тех интернет-проектов, которые ориентированы на молодую и веб-активную часть населения. Например, интернет-магазины, площадки бронирования билетов и гостиниц сильно попортят свой бизнес, если сделают ставку на регистрацию через социальные сети, забыв про традиционный способ. Далеко не все взрослое платежеспособное население располагает такими аккаунтами, а те, кто располагает, не стремятся привязывать покупки к персональным аккаунтам.

Из всего сказанного следует сделать один очевидный вывод: капчи при регистрациях на сайте, конечно, необходимы. Но они должны учитывать особенности человеческого восприятия. Пусть каждый владелец сайта спросит себя, способен ли он понять содержание и порядок знаков, например, в такой капче.



Если нет, то она не имеет права на существование. А все «открытые» капчи с возможностью написания ответа разными способами обязательно должны сопровождаться подсказками.

Закон №2. Требования к сложности пароля должны быть очевидными

Также новых пользователей раздражают сообщения об ошибках при введении нового пароля. Обычно ошибка заключается в том, что пароль содержит недостаточное количество знаков, например, менее 6. Однако на многих сайтах нет подсказок, сколько минимум знаков должен содержать пароль. Отсюда частые сообщения об ошибках и необходимость заново придумывать пароль.

Наиболее известная в России компания в области веб-юзабилити – это, конечно, UsabilityLab. Их сайт – воплощение лозунга дня «Сделай проще». В нем не хватает так любимых многими дизайнерских фенечек, зато здесь достаточно просто ориентироваться. И все же...

Однажды наш коллега попытался зарегистрироваться на сайте, а именно в разделе «Интернет-магазин», где предлагается заполнить форму. Он ее заполнил и получил в ответ следующее.

Главные специалисты по юзабилити, как себя позиционирует эта компания, забыли, что а) обязательные поля принято выделять звездочкой и б) требования к заполнению полей желательно указывать до, а не после того, как пользователь неправильно введет данные.

Я не спешу кричать: «А король-то голый!» Кто не без греха, пусть первый бросит камень. Я всего лишь делаю вывод, что у нас даже «лучшие» (точнее, наиболее раскрученные) специалисты по юзабилити могут совершать элементарные ошибки. Как всегда в нашей стране, раскрутка бежит впереди профессионализма и качества. ■

Новый пользователь

Пароль должен быть не менее 6 символов длиной.

E-mail:

Пароль:

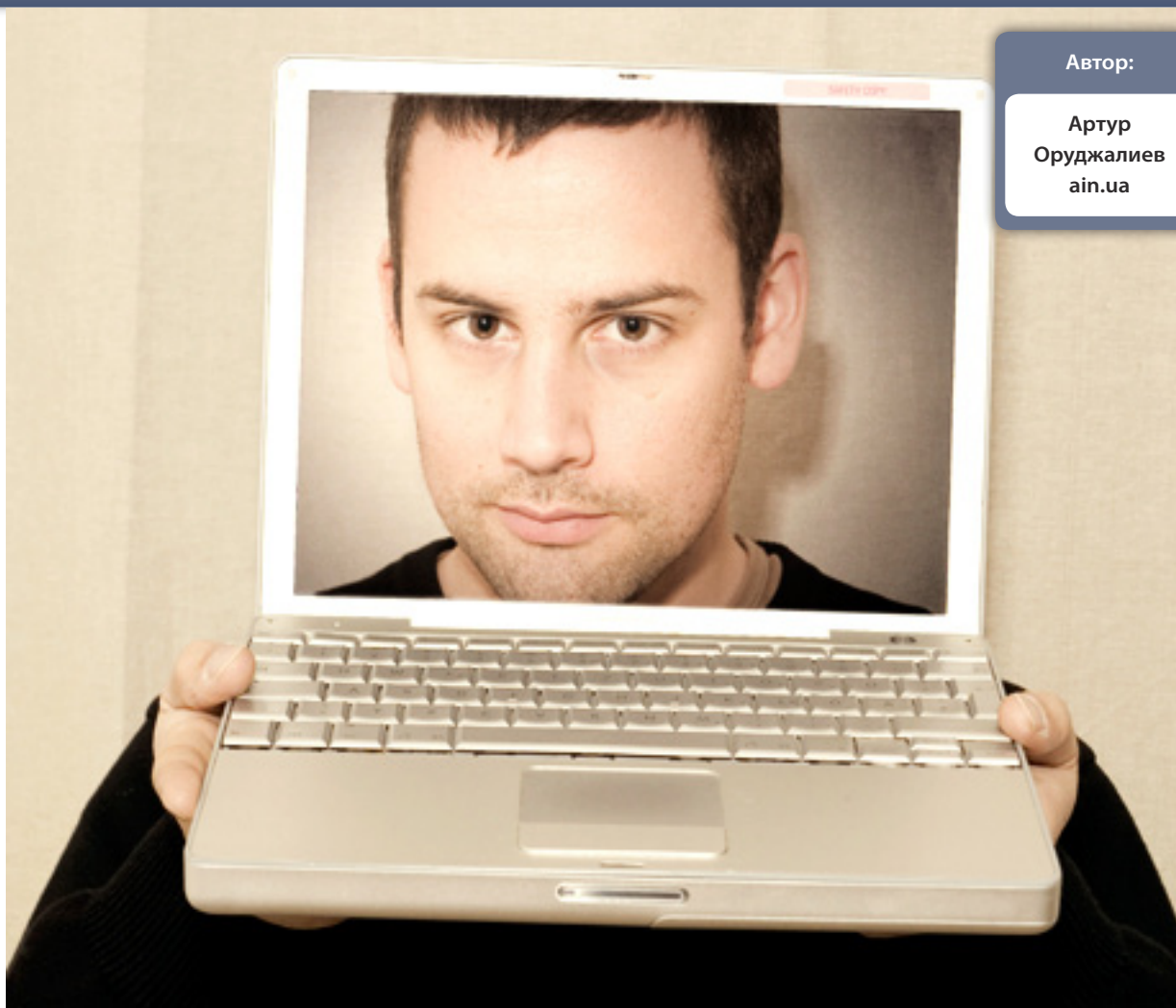
Подтверждение пароля:

Имя:

Фамилия:

Готово

Автор:

**Артур
Оруджалиев**
ain.ua

Завел Facebook – забудь о privacy

Личность любого человека скоро можно будет определить за несколько секунд по его профайлу в социальной сети, показало исследование Университета Карнеги — Меллона

Научные фантасты уже давно рисовали будущее, в котором технология распознавания лиц уничтожит анонимность. Согласно исследованию, проведенному Университетом Карнеги — Меллона, это будущее уже наступило благодаря доступным в продаже продуктам для распознавания лиц в сочетании с базой данных на 750 млн человек под названием Facebook.

Исследовательская команда университета под руководством доцента Алессандро Аквисти с помощью скры-

той веб-камеры стоимостью \$35 сделала фотографии 93 случайно попавших под объектив студентов в кампусе. В течение нескольких секунд исследователям удалось определить личности 1/3 сфотографированных студентов с помощью готового программного обеспечения для распознавания лиц от софтверной компании PittPatt, недавно приобретенной Google, а также фотографий с Facebook, выложенных в общий доступ. Еще больший успех их ждал при использовании той же технологии для определения личности 100

000 одиноких сердец из Питсбурга, зарегистрированных под псевдонимом на сайте знакомств.

«Мы думали о тайне личной жизни в киберпространстве, но пора задуматься о ней в реальном мире, в мире дополненной реальности», — говорит Аквисти. Он также смог определить первые пять цифр номеров социального страхования у некоторых из опознанных студентов, основываясь на предыдущем исследовании, показывавшем, что эти цифры предсказуемы, если знать день и место рождения ►

человека. Эта информация обычно доступна на странице личной информации в Facebook.

Похоже, автор известного романа антиутопии «О дивный новый мир» Олдос Хаксли был прав, а Джордж Оруэлл ошибался. Вездусущий надзор не организуется правительством Большого брата из «1984» — благодаря прорывам в технологии люди сами готовы делиться информацией о себе. Аквисти называет этот процесс «демократизацией надзора». И он быстро развивается. Вскоре после [беспорядков в Лондоне](#) в начале августа добровольцы сформировали группу в Google, которая обсуждала, можно ли с помощью выводов исследовательской команды Аквисти помочь установить личности мародеров по фотографиям и видеосъемке.

Инструменты для распознавания лиц устанавливают личность человека, анализируя десятки физиогномических черт, таких как высота лба и расстояние между глазами и носом. Google, Facebook и Apple уже сделали их доступными бесплатно для указания друзей и родственников в фотоальбомах. Но в какой момент у вас появится возможность сделать фотографию незнакомца и получить его имя и любую другую информацию, имеющуюся в интернете?

Возможность есть, но желания сделать ее доступной нет. Пока. Так, инструмент для поиска картинок [Goggles](#) от компании Google, который, например, позволяет снять картину и выяснить ее создателя, способен установить личность человека по его фотографии. Но поисковый гигант предпочел не запускать эту функцию, посчитав ее слишком «страшной», признался недавно председатель совета директоров Google Эрик Шмидт.

Когда Facebook недавно выпустила внутренний инструмент для распознавания лиц, чтобы автоматически выделять друзей на фотографиях, это вызвало протест некоторых защитников тайны частной жизни (которые имеют тенденцию яростно осуждать все новое на Facebook). Эти защитники, похоже, упускают из виду тот факт, что само существование обширной базы Facebook с фотографиями, прикрепленными к именам людей, дает третьим лицам возможность строить биометрические снимки. Хотя формально

Facebook запрещает использовать информацию со своего сайта, для предотвращения этого нет никаких технологических средств. И даже если бы они были, информацию можно получить не только в Facebook. «У большинства из нас есть фотографии с подписью где-то в интернете, и эта тенденция будет только усиливаться», — говорит Аквисти.

Исследователи из Университета Карнеги-Меллона разработали приложение для распознавания лиц для iPhone, но решили не выкладывать его в общественный доступ. «Мы разработали это приложение, чтобы доказать свои выводы, — говорит Аквисти. — Будущее, в котором можно получить конфиденциальную информацию по изображению лица, уже наступило, и нам лучше задуматься, действительно ли мы хотим жить в таком будущем».

Инструмент исследователей, однако, будет ценен только для тех, кто пытается идентифицировать кого-либо из 25 000 студентов Университета Карнеги-Меллона, чьи фотографии были взяты с Facebook. От приложения, которое бы работало в государственном или мировом масштабе, нас отделяет еще несколько лет. Две главные трудности, по словам Аквисти, это найти крупный и надежный источник изображений лиц (здравствуй, транспортный департамент!) и обеспечить достаточную вычислительную мощность. На сравнение фотографии с базой, скажем, из 300 млн изображений уйдет много часов, а хорошо работающее приложение должно делать это за 10 секунд. «Если бы меня попросили угадать, я бы сказал, что нам осталось до этого лет 5», — говорит Аквисти.

Специальные технологии для распознавания лиц уже доступны на рынке. Компания BI2 Technologies разработала приложение для iPhone за \$3000 под названием MORIS (Mobile Offender Recognition and Identification System — Мобильная система распознавания и идентификации правонарушителей), которое в этом году начали использовать десятки полицейских управлений по всей Америке. Офицеры полиции с помощью MORIS могут фотографировать подозреваемых, и приложение, используя систему распознавания лиц и узора радужной оболочки, установит личность любого, чья

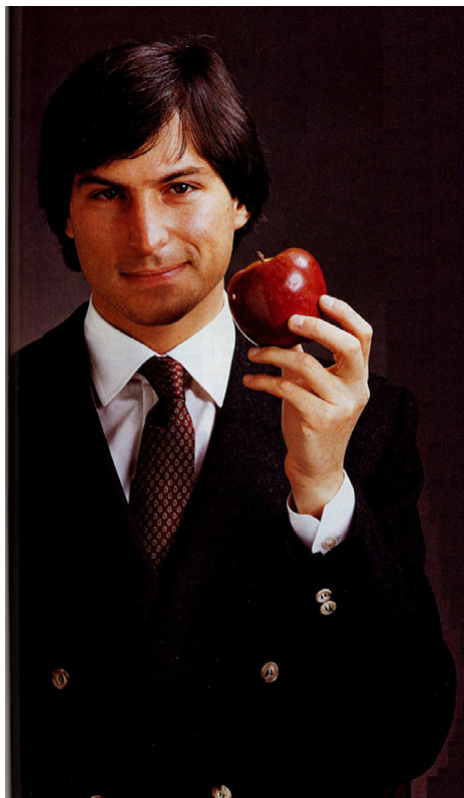
фотография уже имеется в базе преступников.

Microsoft вместе с Intel создали цифровые рекламные щиты с камерами, распознающими лица, чтобы определять пол и возраст прохожих и показывать им подходящую рекламу. Стартап под названием SceneTap в этом году установил подобные камеры в барах Чикаго, чтобы любители ночной жизни Города ветров могли с помощью своих iPhone проверять наполненность, половой состав и возраст посетителей бара, прежде чем принять решение, куда пойти.

На данный момент служащие правоохранительных органов с осторожностью относятся к использованию технологий для установления личности людей, еще не включенных в базу преступников. Хотя одна канадская страховая корпорация великодушно предоставила свою базу фотографий с водительских удостоверений полиции Ванкувера после разрушительных хоккейных беспорядков, произошедших там в июле после матча на Кубок Стэнли, власти отказались ею воспользоваться, когда правозащитники выразили озабоченность по поводу нарушения гражданских свобод канадцев. В Лондоне Скотланд-Ярд использует инструменты распознавания лиц наряду с другой помощью от сообщества для идентификации бунтовщиков.

Как бы хороша ни была технология, до сих пор остаются проблемы с точностью, и они могут причинить ощутимый вред. Водитель из Массачусетса подал в суд на транспортный департамент штата за то, что его права были по ошибке аннулированы из-за его схожести с другим водителем, после чего ему пришлось 10 дней воевать с чиновниками, чтобы их восстановить. «Мы считаем себя красивой и неповторимой снежинкой, но существует множество людей, похожих на нас», — говорит Аквисти. Транспортные департаменты в трех десятках штатов уже применяют алгоритмы распознавания лиц к фотографиям на водительских удостоверениях для предотвращения мошенничества.

Если вы беспокоитесь о тайне своей личной жизни, возможно, сейчас самое время заменить фотографию в Facebook на ту, где вы в огромных очках. ■



Автор:

Линдер Кани
www.cultofmac.com

Маркетинг Стива Джобса

Джон Скалли хорошо известен знатокам истории компании Apple и интересующимся фигурой Стива Джобса — Скалли был первым правильным CEO Apple Computer, именно его Стив Джобс переманил из Pepsi фразой про торговлю сладкой водичкой, именно он возглавил компанию после того, как Джобса ушли из компании, и развивал ее 10 лет, сражаясь с превосходящими силами РС-клонов.

Совсем недавно сайт Cult of Mac взял, пожалуй, первое подробное интервью у Джона Скалли, в котором обсуждается не история Apple, а исключительно сам Стив Джобс, стиль его работы, подход к проектированию продуктов и их маркетингу. Ниже приводится полный перевод интервью, которое очень уместно прочесть в преддверии очередной презентации Apple, с нетерпением ожидаемой всей армией маководов.

Вопрос: Вы говорите о «методологии Стива Джобса». Что это за методология?

Скалли: Попробую дать вам общее представление. Впервые я встретил Джобса более 25 лет назад. В это время он как раз собирал в единую концепцию те самые принципы, которые я называю методологией создания отличных продуктов Стива Джобса.

С момента нашей встречи Стив всегда любил красивые вещи, особенно различные технические приспособления. Придя однажды ко мне домой, он был очарован специальными петлями и замками, висевшими на моих дверях. Я учился на промышленного дизайнера, и именно промышленный дизайн стал той вещью, которая связала нас со Стивом. Вовсе не вычислительная техника.

В действительности, я не знал вообще ничего о компьютерах, как и любой другой человек в мире в то время. Революция в области персональных компьютеров только начиналась, но мы оба верили в красивый дизайн, и Стив был в особенности уверен, что начинать разработку следует, отталкиваясь от восприятия продукта пользователем.

Какое впечатление пользователь

получит от продукта? Стив всегда смотрел на вещи с этой точки зрения. Многие люди, занимавшиеся маркетингом продуктов в те дни, выходили «в поля» и исследовали потребительские предпочтения, спрашивая людей о том, что они хотят; но Стив не верил в этот метод.

Он говорил: «Как можно спрашивать кого-то о том, каким должен быть компьютер с графическим интерфейсом, если они не представляют, что вообще такое графический интерфейс? Его никто никогда раньше не видел». Он считал, что если показать кому-то, например, калькулятор, это не позволит ничего понять о перспективах компьютера — слишком велика разница между двумя устройствами.

Стив всегда придавал большое значение восприятию продукта пользователем, и промышленный дизайн был невероятно важной частью этого впечатления. И он привлек меня к работе в Apple, потому что верил — в конечном итоге, компьютер станет потребительским товаром. Тогда, в начале 1980-х, это была чрезвычайно странная идея, потому что люди считали персональные компьютеры просто уменьшенными копиями больших компьютеров. Именно так полагали в IBM.

Некоторые считали, что персональ-

ный компьютер больше похож на игровую приставку — в то время как раз начали появляться игровые приставки, они были очень простыми и подключались к телевизору... Но Стив думал о совсем других вещах. Он чувствовал, что компьютер вот-вот изменит мир и станет тем, что он называл «велосипедом для ума». Это позволит каждому человеку получить невероятные возможности, о которых он никогда не мечтал раньше. Это не имеет никакого отношения к игровым приставкам. И никакого отношения к большим компьютерам уменьшенного размера...

Стив имел огромный дар предвидения. В то же время, он придавал большое значение точному планированию каждого шага. Он был методичен и аккуратен во всем перфекционист от начала и до конца.

Возвращаясь к Apple II, Стив был первым, кто поместил компьютер в пластиковый корпус из материала, который в те дни назывался АБС-пластик, и фактически укомплектовал компьютер клавиатурой. Сегодня это выглядит довольно простой идеей, но в то время, когда создавался первый Apple II, в 1977 году — это стало началом методологии Джобса. Методологии, которая проявила себя в Macintosh, в компьютере NeXT, а затем — в Mac-ax, ►

iMac-ax, iPod-ax и iPhone-ax будущего.

Вот еще кое-что, что отличает методологию Стива от всех остальных. Он всегда считал, что наиболее важные решения — это не то, что вы делаете, но то, что вы решили не делать. Он минималист.

Я помню, как однажды пришел в дом к Стиву. У него там не было даже мебели — только фотография Эйнштейна, которым он всегда восхищался, лампа в стиле Тиффани, стул и кровать. Он никогда не стремился иметь много вещей вокруг себя и был невероятно аккуретен в их выборе. Так же получилось и с Apple. Стив сделал ставку на то, как продукт воспринимается пользователем, он верил, что промышленный дизайн нельзя сравнивать с чем-то иным, что люди делают технологичными продуктами, кроме как с работой над ювелирным украшением... Можно вернуться к моему примеру с замком, петлями и дверью с красивыми, медными, качественно изготовленными механическими устройствами. Я думаю, это является характерной чертой всего, к чему когда-либо приложил руку Стив.

Когда я впервые увидел Macintosh — он тогда был в процессе создания — это был лишь набор компонентов, макет электронной схемы. Считайте, что не было еще ничего. Но Стив всегда обладал способностью находить самых лучших и умных людей, которые могли, по его мнению, быть поблизости. Он обладал невероятной харизмой и легко убеждал людей присоединиться к нему. Он умел вдохновить людей своим видением продукта еще до его появления. Когда я увидел команду Мас, она насчитывала около ста человек. Но в то время, когда я только встретил Стива, у него было куда меньше сотрудников. И их средний возраст не превышал 22 лет.

Это были люди, которые до этого никогда не занимались разработкой коммерческих продуктов. Но они верили в Стива и в его видение. Он мог параллельно работать на нескольких уровнях.

На одном уровне он работал над большой идеей «изменения мира». На другом — углублялся в разработку деталей продукта, в дизайн программного обеспечения, аппаратного обеспечения и систем, а также в приложения и периферийные продукты.

В каждом случае, он всегда окружал себя самыми лучшими людьми, которых только мог найти. И всегда лично проводил отбор сотрудников в свою команду. Он никогда не отдавал это дело кому-либо другому.

Еще одна деталь о Стиве — он никогда не уважал большие организации. Он считал, что они бюрократичны и неэффективны. Он всегда называл так организации, к которым не испытывал уважения.

Команда Мас находилась вся в одном здании и, в конечном итоге, выросла до ста человек. У Стива было правило, что в команде Мас никогда не будет больше ста человек. Так что если вам хотелось взять кого-либо еще, то приходилось увольнять одного из сотрудников. И такой подход Стив основывал на простом наблюдении: «Я не могу запомнить больше ста имен, так что я лишь хочу работать с теми людьми, которых знаю лично. Поэтому если в моей команде будет больше ста человек, нам придется выстраивать работу в организации по-другому, и я не смогу больше работать, как мне нравится. А мне нравится работать так, чтобы я мог участвовать во всем». Все время, что я знаю его в Apple, именно так он и работал.

Вопрос: Так как же тогда он справился с ситуацией, когда Apple стала больше? Ведь сейчас компания насчитывает десятки тысяч сотрудников.

Скалли: Стив бы сказал: «Организация может стать больше, но не команда Мас». Macintosh был утвержден в качестве отдела разработки продукта — при этом у компании Apple был центральный отдел, занимающийся продажами, центральный офис для всех административных и правовых вопросов. Также здесь был отдел, занимающийся производством. Но именно команда Мас была той командой, которая разрабатывала продукт. Когда ты занимаешься разработками в сфере высоких технологий, тебе совсем не нужно большое количество людей, чтобы сделать действительно хороший продукт. Скорее всего, здесь вы увидите лишь небольшую группу программистов, которые заняты созданием операционной системы. Люди думают, что над ее

созданием должны работать сотни сотрудников. Но это не так. На самом деле, нужна лишь небольшая команда людей. Думайте об этом, как об ателье. Это действительно похоже на мастерскую, где работают художники. И Стив здесь как мастер, который ходит вокруг, смотрит на их работу и делает замечания, требует убрать что-то.

Я помню множество вечеров, когда мы могли задерживаться на работе далеко за полночь, потому что инженеры были обычно не слишком активны до обеда, но зато отлично работали ближе к ночи. И инженер мог подозвать Стива и показать ему последний программный код, который он написал. Стив же мог посмотреть все и бросить обратно разработчику со словами: «Это недостаточно хорошо». Он постоянно заставлял людей повышать их ожидания от того, что они могут сделать. И его сотрудники делали работу, на которую раньше не считали себя способными. По большому счету, это ему удавалось благодаря умению переключаться. С одной стороны, он мог быть харизматичным, давать людям ощущение, что они являются частью чего-то невероятного и великого. И в то же время, он мог беспощадно критиковать их работу, пока не чувствовал, что добился совершенства, и что теперь эта деталь может пойти — в данном случае — в Macintosh.

Вопрос: Он хорошо осознал, что делает, верно? Это была очень хорошо продуманная политика, а не просто капризы сумасшедшего?

Скалли: Нет, Стив был невероятно методичен. У него в офисе всегда была белая чертежная доска. Но он никогда не рисовал сам. У него не было особых способностей к рисованию, но в то же время был прекрасный вкус.

Это та вещь, которая отделяет Стива Джобса от других людей, таких как Билл Гейтс. Билл тоже был выдающимся талантом, но он никогда не интересовался вопросами вкуса. Он всегда интересовался возможностью доминировать на рынке. Он бы предпринял всё возможное, чтобы завоевать это пространство. Стив никогда не сделал бы этого. Стив верил в совершенство. Стив был готов на необычайный риск в попытке захватить новую продук-

товую нишу, однако всегда смотрел на этот риск с точки зрения дизайнера. Бывают разные виды руководителей — одни руководители являются сильными лидерами, другие лучше всего умеют справляться с кризисными ситуациями, третьи прекрасно ведут переговоры, четвертые умело мотивируют людей. Что касается Стива, его самой сильной стороной был дизайнерский талант. Все в Apple можно лучше всего понять, рассматривая через призму дизайна.

Будь то проектирование внешнего вида и сценариев взаимодействия интерфейса с пользователем, или промышленный дизайн, или системное проектирование и даже такие вещи, как расположение монтажных плат. По мнению Стива, платы должны были выглядеть красиво, когда вы смотрите на них, несмотря на то что, создавая Macintosh, он лишил пользователя возможности заглянуть внутрь корпуса, потому что не хотел позволить им вносить самовольные изменения в конструкцию.

Его уровень перфекционизма требовал, чтобы каждая деталь выглядела прекрасно, даже если большинство людей никогда ее не увидит.

Стремление к совершенству перешло на уровень производственного оборудования, когда он построил завод по производству компьютеров Macintosh. Это должен был быть полностью автоматизированный завод, но в действительности это был цех окончательной сборки и тестирования, оснащенный роботизированной упаковочной линией. Сегодня это звучит не так инновационно, как 25 лет назад, но я помню, как генеральный директор General Motors вместе с Россом Перо приехали просто посмотреть на завод Macintosh. Все, чем мы занимались это окончательная сборка и тестирование, но до чего же красиво был реализован этот нехитрый производственный процесс! Дизайн этой автоматизированной фабрики, на которой в безлюдную ночную смену выключалось освещение, был продуман так же глубоко, как и дизайн собираемых на ней продуктов.

Переместимся в наше время и посмотрим на продукты, которые Стив создает сегодня. Современные технологии дают намного больше возможностей, они позволяют провести миниа-

туризацию, они превратились в товар, они недороги. И Apple больше ничего не производит. Когда я был там, люди называли Apple «вертикально-интегрированным рекламным агентством», и это был не комплимент.

В действительности, сегодня все используют эту модель. Ее использует HP, ее использует Apple и большинство других компаний, потому что они привлекают в качестве подрядчиков компании, производящие электронику.

Вопрос: Nike — это хорошая аналогия?

Скалли: Да, наверное, Nike близок к этому, думаю, что это так. Я думаю, что если вы посмотрите на японских производителей бытовой электроники той эпохи, то обнаружите, что все они были аналоговыми компаниями.

Компанией, которой Стив восхищался, была Sony. Мы регулярно беседовали с Акио Морита он действительно ценил красивые продукты и придерживался тех же стандартов высокого класса, что и Стив. Я помню, как Акио Морита подарил Стиву и мне по плееру Sony Walkman, это была одна из первых моделей. Никто из нас никогда не видел ничего подобного раньше, потому что никогда раньше не было такого продукта, как этот. Это было 25 лет назад, и Стив был очарован этим плеером. Первое, что он сделал со своим разобрал его и изучил каждую деталь. Как были подогнаны друг к другу компоненты, как была выполнена сборка.

Он был очарован заводами Sony. Мы прошли их насквозь. Они были полны людей в разноцветной униформе. У кого-то была красная униформа, у кого-то зеленая, у кого-то синяя, в зависимости от того, каковы были функции данного работника. Все было тщательно продумано, заводы были безупречны. Все эти вещи производили огромное впечатление на него.

Завод Мас был устроен именно так. Хотя униформа рабочих не была разноцветной, она была столь же элегантна, как и на заводах Sony, которые мы видели ранее. Ориентиром Стива в то время была Sony. Он действительно хотел быть Sony. Он не хотел быть IBM. Он не хотел быть Microsoft. Он хотел быть Sony.

Проблема была в том, что в ту эпоху

вы не могли бы создавать цифровые продукты подобно Sony. Все было аналоговым, достижения японских компаний были впечатляющими.

Sony должна была разработать iPod, но она не сделала этого — а сделала Apple. iPod является прекрасным примером методологии Стива начать с пользователя и продумать всю систему от начала и до конца.

У Стива все системы были продуманы от начала и до конца. Он не был дизайнером, но имел великолепное системное мышление. Это то, чего не хватает другим компаниям. Они склонны сосредотачиваться на одном участке работы и отдавать на сторону все остальное.

Если вы посмотрите на iPod, цепочка поставок, продолжающаяся до самого города iPod в Китае, будет такой же сложной, как дизайн самого продукта. К цепочке поставок применяются стандарты совершенства, настолько же серьезные, что и стандарты, применяемые к пользовательскому дизайну. Это совершенно другой взгляд на вещи.

Вопрос: Откуда у него эта идея — контролировать все самому? Идея отвечать за все, за систему целиком?

Скалли: Стив считает, что если поделиться частью ответственности с другими, люди начнут вносить небольшие изменения в систему, эти изменения будут компромиссными решениями, и он не сможет добиться такого восприятия продукта пользователями, какого хотел бы добиться.

Вопрос: Но этот контроль распространяется на все аспекты продукта — даже на открытие коробки. То, каким должно быть впечатление от открытия коробки, продумано лично Стивом Джобсом.

Скалли: Оригинальный Мас действительно не имел операционной системы. Люди постоянно говорили: «Почему бы нам не лицензировать операционную систему?» Простой ответ не было ни одной подходящей. В то время, чтобы заставить аппаратное и программное обеспечение работать, требовалось большое количество трюков. Микропроцессоры в те дни ►

были очень слабы по сравнению с тем, что мы имеем сегодня. Простой вывод графики на экран потреблял всю мощность процессора. Поэтому было необходимо наклеить вокруг процессора другие чипы, чтобы выгрузить в них остальные функции. Затем нужно было установить то, что называется «вызовы ROM». Там было 400 вызовов ROM; все они были небольшими подпрограммами, которые нужно было выгрузить в ROM, поскольку не было никакой возможности запускать их в режиме реального времени. Все эти вещи были аккуратно скреплены вместе. Это было совершенно замечательно — продавать такие компьютеры, в то время как первый процессор на Mac выполнял менее трех MIPS (миллионов инструкций в секунду). Сегодня это все равно что... нет, я не могу вспомнить ни одного насколько медленного устройства, которое бы выполняло три MIPS или около того. Даже ваши цифровые часы, по крайней мере, в 200 или в 300 раз более мощны, чем первый Macintosh. (Примечание: Для сравнения, современный iMac начального уровня использует процессор Intel Core i3 мощностью более 40000 MIPS!)

Трудно представить, каким образом в те дни он смог добиться так многого, имея такие ограниченные ресурсы. Повторить наш успех с первым Mac'ом в 1980-х годах не мог ни один производитель потребительских товаров. В 1990-х годах закон Мура и другие вещи, такие как гомогенизация технологий, дали возможность увидеть то, как могли бы выглядеть потребительские товары, даже если вы не можете реально сконструировать их. До начала нового века действительно не было возможности выбирать нужное вам сочетание стоимости компонентов, коммодитизации и миниатюризации. Производительность вдруг достигла точки, в которой действительно можно создавать то, что мы называем цифровыми потребительскими товарами. Методология дизайна Стива была более чем верна. Уже 25 лет назад он был в состоянии разработать методологию дизайна свои основные принципы — методологию, которая требует опираться на впечатления пользователя, сосредотачиваться на нескольких вещах, обозревать систему в целом,

не идти на компромиссы, сравнивать себя не с другими электронными продуктами, но с лучшими ювелирными изделиями. Но никто больше не думал об этом. Все остальные просто пытались воспользоваться эволюцией дешевых продуктов, которые становились все более мощными и недорогими в разработке. Таких продуктов, как MP3-плеер. Помните, когда он пришел с iPod, на рынке были тысячи MP3-плееров. Может сейчас кто-нибудь вспомнить название любого из них?

Побочным эффектом была его вера в то, что он должен лично контролировать всю систему. Он принимал каждое решение. Корпуса были закрыты.

Вопрос: Но мотивацией для этого было восприятие продукта пользователем?

Скалли: Совершенно верно. Восприятие продукта пользователем должно отразиться на всей системе, будь то издательская программа или iTunes. Все это — неотъемлемая часть системы. Кроме того, это производство. Цепочка поставок. Маркетинг. Магазины. Я помню, что был принят в команду, потому что имел дизайнерское образование, и еще, потому что я был маркетологом. У меня был опыт маркетинга продуктов. Не потому, что я что-то знал о компьютерах.

Вопрос: По-моему, это весьма интересно. В вашей книге вы говорите, что в первую очередь хотели сделать Apple «компанией продуктового маркетинга».

Скалли: Верно. Стив и я потратили несколько месяцев, чтобы узнать друг друга поближе, прежде чем я присоединился к Apple. Он не имел никакого представления о маркетинге, кроме того, что понял самостоятельно. Это типично для Стива. Когда он знает, что что-то будет иметь большое значение, он пытается понять это настолько глубоко, насколько может.

Одна из вещей, которые приводили его в восторг: Я рассказал ему, что между Pepsi и Coke не так много разницы, но наши продажи были лучше в соотношении 9 к 1. Наша задача была убедить людей, что перейти на Pepsi

— это достаточно важное решение, что они должны обратить внимание на свой выбор, и в конечном итоге переключиться. Мы решили, что создать к Pepsi такое же отношение, как к галстуку. В то время люди обращали внимание на то, какой галстук они носят. Галстук говорил: «Я хочу, чтобы вы видели меня таким». Так что нам было необходимо превратить Pepsi в хороший галстук. Когда вы держите Pepsi в руке, это говорит: «Я хочу, чтобы вы видели меня таким».

Мы провели ряд исследований и обнаружили, что когда люди собираются угостить друзей у себя дома безалкогольными напитками, если у них в холодильнике лежит Coca Cola, они пойдут на кухню, откроют холодильник, вынут бутылку, принесут ее, поставят на стол и нальют стаканы перед своими гостями.

Люди, которые имели в холодильнике Pepsi, выходили на кухню, вынимали бутылку из холодильника, открывали ее, наполняли стаканы на кухне, и только потом приносили их гостям. Дело в том, люди стеснялись того, что они угощают Pepsi, и не хотели, чтобы кто-то об этом знал. Возможно, они бы подумали, что это была Coca Cola, потому что Coca Cola производила лучшее впечатление. Это был более хороший галстук. Стив был очарован всем этим.

Мы много говорили о том, как впечатление определяет действительность и о том, что вы должны быть в состоянии создать впечатление, если собираетесь создавать действительность. Мы уже сделали это с тем, что называется поколением Pepsi.

Мы фокусировали свой маркетинг на многих вещах, которые делали, когда выводили Mac на рынок. Мы обеспечивали такой высокий уровень ожидания, что Стив как будто бы дразнил людей, которые хотели узнать, на что способен наш продукт. Но поначалу продукт не очень-то многое мог. Почти все технологии использовались для того, чтобы улучшить восприятие продукта пользователями. Мы, в самом деле, получили частичную негативную реакцию пользователей, люди посчитали наш продукт игрушкой. С его помощью ничего нельзя было сделать. Но, в конце концов, по мере развития технологий, стало можно сделать все. ►

Вопрос: Конечно, сегодня Apple по-прежнему рекламирует стиль жизни. Эта реклама показывает людей, живущих завидной жизнью благодаря продуктам Apple. Модно одетые молодые люди, наслаждающиеся музыкой из iPod'ов...

Скалли: Я не имею никакого отношения к этому. Гениальность Стива заключается в его способности увидеть что-то, потом понять это, а затем придумать, как включить эту вещь в контекст его методологии дизайна. Дизайн — это всё.

Расскажу одну историю. Один мой друг побывал на совещаниях в Apple и Microsoft, которые проходили в тот же день. Это было в прошлом году, совсем недавно. Он пришел на совещание Apple (он поставщик Apple), и перед началом этого совещания, как только дизайнеры зашли в комнату, все замолчали, потому что дизайнеры — это самые уважаемые люди в организации. Всем известно, что дизайнеры говорят от имени Стива, потому что они напрямую подчиняются ему. Только в Apple дизайн подчиняется непосредственно генеральному директору.

Позже в тот же день он был в Microsoft. Когда он пришел на совещание в Microsoft, все говорили, а затем совещание началось, и ни один дизайнер не вошел в комнату. Там сидели технари и пытались выдвигать свои идеи о том, каким должен быть дизайн. Это путь к катастрофе.

Microsoft нанимает некоторых из самых умных людей в мире. Известно, что они подвергают невероятно сложным испытаниям людей, которые пытаются устроиться к ним на работу. Так что здесь речь не о том, насколько люди умны и талантливы. Она о том, что дизайн в Apple находится на самом высоком уровне организации, во главе со Стивом лично. Дизайн в других компаниях не существует. Он похоронен где-то посреди бюрократии... В бюрократических организациях многие люди имеют право сказать «нет», но не право сказать «да». И вы в конечном итоге получаете компромиссный продукт. Это восходит к философии Стива, которая гласит, что наиболее важные решения это то, что вы решили НЕ делать, а не то, что вы решили делать. Это снова минималистское мышление.

Я был рядом почти с самого начала, и с тех пор я не заметил каких-либо изменений в основных принципах Стива за исключением того, что он все более и более последователен в них.

Другой блестящий пример это то, что он сделал с розничными магазинами.

Он принял в команду одного из лучших торговцев в мире, чтобы научиться у него розничным продажам (это был Микки Дрекслер из The Gap, который посоветовал Джобсу создать прототип магазина перед запуском). Он не просто научился розничной торговле, я никогда не видел магазина лучше, чем магазин Apple. Он получает более высокий доход на квадратный фут, чем любой другой магазин мира, но дело не только в доходе. Дело во впечатлении.

Магазины Apple наполнены людьми. Вы можете пойти в центр Sony — например, в центре Москони в Сан-Франциско. Там никого нет. Вы можете пойти в магазин Nokia, у них есть один в Нью-Йорке на 57-й улице. Там тоже никого нет.

Но вообще-то люди ходят по магазинам. У них есть продукты, на которые они хотели бы посмотреть. Вы можете коснуться и почувствовать их, но вот вы входите в магазин Apple и это потрясающее впечатление. Там так много людей, которые совершают покупки вместе с вами.

Опять же, это продукт, подобный галстуку. Когда вы находитесь в магазине Apple, вы как бы говорите окружающим: «Я хочу, чтобы вы меня видели таким. Я здесь. Я среди гениев. Я изучаю эти продукты. Посмотрите на меня: Я такой же, как остальные люди в этом магазине».

Впечатления пользователей фиксируются на каждом этапе от непосредственного использования продукта, до рекламы и дизайна. Жесткие требования Стива к качеству сборки продукта приобрели легендарную известность. Поглядите на эти радиусы, стыки и грани все эти маленькие детали, на которые дизайнеры обратили внимание.

Он будет отклонять то, в чем никто не увидит проблему. Но потому, что его стандарты настолько высоки, люди сидят там и говорят: «Как Apple делает это? Как Apple удается создавать такие невероятные продукты?»

Вопрос: Давайте поговорим о рекламе, которая так важна для Apple. В своей книге вы говорите о «стратегической рекламе» реклама как стратегия. Это очень интересная идея...

Скалли: В то время, когда я приехал в Силиконовую долину, там не было рекламы... Единственной компанией, действительно заинтересованной в рекламе, была Apple. HP не рекламировались в те дни. Тогда никто не рекламировал свой основной бренд. Я был принят на работу в Apple, чтобы помочь сделать несколько вещей, и одной из этих вещей был запуск большой рекламной кампании бренда Apple.

Логотип Apple был цветным, поскольку Apple II стал первым компьютером, который поддерживал отображение различных цветов. Больше никто не мог работать с цветом, поэтому они поместили цветные блоки в логотип. Если вы хотели напечатать логотип в рекламе в журнале или на пакете, вы могли распечатать его в четырех цветах, но Стив не был бы Стивом, если бы не настоял на шести цветах. Поэтому, где бы ни размещался логотип Apple, он всегда был напечатан в шести цветах. Это добавляло еще от 30 до 40 процентов к стоимости всего, но так хотел Стив. Это то, что мы всегда делали. Он был перфекционистом с самых первых дней.

Вопрос: Многих людей такие причуды могут довести до белого каления. Вас когда-нибудь доводили?

Скалли: Это нормально, когда тебя доводит до белого каления человек, который постоянно оказывается прав. В сфере высоких технологий я выяснил, что грань между успехом и провалом очень-очень тонка. Это отрасль, где вы постоянно рискуете, особенно если вы такая компания, как Apple, которая постоянно живет на краю.

Ваши шансы быть по одну сторону от этой линии или по другую сторону от нее примерно равны. Иногда ... он ошибался тактически в ряде вещей. Он не согласился установить жесткий диск в Macintosh. Когда кто-то спросил его об обмене информацией, он просто бросил дискету через комнату и сказал: «Это все, что нам когда-либо ►

будет нужно». С другой стороны, Стив возглавил разработку продуктов, которые назывались AppleTalk и AppleLink. AppleTalk был интерфейсом, позволившим связать Macintosh с лазерным принтером, что положило начало... настольным издательским системам.

AppleTalk был выдающимся достижением своего времени. Он был настолько же выдающимся как Macintosh. Это был еще один пример использования минималистского подхода и решения проблемы, в то время, когда никто не подозревал, что эта проблема существует. Стив еще в 80-х годах решал проблемы, и 15, 20 лет спустя оказалось, что это были именно те проблемы, которые стоило решать. Сложность была в том, что до момента, когда технология станет достаточно стандартизированной и достаточно мощной, чтобы позволить вывести все эти вещи на массовый рынок, оставалось еще несколько десятилетий. Во многих случаях он просто далеко опережал свое время.

Оглядываясь назад, я вижу, что идея нанять меня в качестве CEO была большой ошибкой. Я не был первым человеком, которого Стив хотел видеть в качестве директора. Он сам был главной кандидатурой, но совет директоров не был готов сделать его CEO, в то время, когда ему было 25 или 26 лет.

Они исчерпали все очевидные кандидатуры на пост CEO из сферы высоких технологий... В конце концов, Дэвид Рокфеллер, который был акционером Apple, предложил поискать CEO в других отраслях и обратиться к лучшему охотнику за головами в Соединенных Штатах: Джерри Рочу.

Они пошли и наняли меня. Я пришел в компанию, не зная ничего о компьютерах. Идея заключалась в том, чтобы Стив и я работали как партнеры. Он был бы специалистом по технике, а я по маркетингу.

Причина, почему я сказал, что нанять меня в качестве CEO было ошибкой, это то, что Стив всегда сам хотел быть CEO. Было бы гораздо честнее, если совет директоров сказал: «Давайте придумаем, каким образом он может стать CEO. Ты сможешь сосредоточиться на своей сфере, а он — на своей».

Вспомните, он был председателем совета директоров, крупнейшим акционером, а еще он возглавлял под-

разделение Macintosh, так что он был одновременно и выше и ниже меня по служебной лестнице. Это было в некотором роде показухой. Я думаю, что мы никогда не пришли бы к разрыву, если бы совет лучше продумал не только то, как можно получить CEO, который бы пришел и возглавил компанию, и которого бы одобрил Стив, но и то, как нам убедиться в том, что созданная нами ситуация будет успешно развиваться с течением времени?

Мне кажется, что когда Стив ушел (в 1986 году, после того, как совет отклонил его предложение заменить Скалли в качестве CEO), я все еще не знал очень многого о компьютерах.

Я решил, что нужно сперва привести компанию в порядок, но я не знал, как приводить компании в порядок и как делать их снова успешными.

Все, что мы делали тогда, было продолжением его идей. Я понял его методологию. Мы никогда ее не изменяли. Поэтому мы не лицензировали продукты. Мы сосредоточились на промышленном дизайне. Мы фактически создали свою собственную проектную организацию, которая и по сей день существует внутри Apple. Мы разработали PowerBook... Мы разработали QuickTime. Все эти вещи были построены на философии Стива... Вся суть была в продажах, маркетинге и эволюции продуктов.

Все дизайнерские идеи определено принадлежали Стиву. Тот, кому действительно стоит отдать должное за все, что происходило, пока я был там, на самом деле Стив.

Я сделал две действительно дурацкие ошибки, о которых я действительно сожалею, потому что думаю, что без них в Apple многое могло пойти иначе. Одна из этих ошибок произошла, когда мы находились в конце жизненного цикла процессора Motorola... Мы создали команду из двух наших лучших технологов, которая должна была изучить вопрос и рекомендовать, что мы должны делать.

Они вернулись и сказали, что не имеет никакого значения, какую RISC-архитектуру вы выберете, нужно просто выбрать ту, которая, как вы думаете, позволит создать лучшие бизнес-решения. Но не нужно использовать CISC. CISC — это процессор с полным набором команд. RISC это компьютер с со-

кращённым набором команд.

Так Intel упорно пытался повлиять на нас, чтобы заставить остаться с ними ... (но) для разработки PowerPC мы заключили альянс с IBM и Motorola. И, как теперь стало ясно, это было ужасное решение. Если бы мы смогли работать с Intel, то получили бы более стандартизированную компонентную платформу для Apple, которая дала бы огромный эффект в течение 1990-х годов. В 1990-е годы процессоры постепенно становились достаточно мощными, чтобы на них могли работать все технологии и программное обеспечение. Как раз в это время Microsoft стартовала со своей Windows 3.1.

До этого вы должны были заниматься как программным, так и аппаратным обеспечением, Apple так и поступала. Когда процессоры стали достаточно мощными, они превратились в стандартный потребительский товар, и все эти подпрограммы, которые мы должны были обрабатывать на аппаратном уровне, стало возможным включить в программное обеспечение.

Таким образом, мы безнадежно опоздали на поезд. Intel вскоре потратит 11 миллиардов долларов и научит свои процессоры работать с графикой... так что это было ужасное решение с технической точки зрения. К сожалению, я не разбирался в технике, поэтому я ушел, получив рекомендации.

Другая, еще большая неудача с моей стороны была в том, если бы я лучше подумал, то вернулся бы к Стиву.

Я хотел уйти из Apple. По прошествии 10 лет, я не хотел оставаться дольше. Я хотел вернуться на восточное побережье. Я сообщил совету, что хочу уйти, к тому же в это время меня пытался завербовать IBM. Они попросили меня остаться. Я остался, но чуть позже они уволили меня. Я действительно не хотел больше там работать.

Совет решил, что мы должны продать Apple. Так что я получил задание уйти и попытаться продать Apple в 1993 году. Так что я ушел и попытался продать компанию AT&T, IBM и другим людям. Мы не смогли найти никого, кто хотел бы ее купить. Они считали это слишком высоким риском, поскольку в то время процветали Microsoft и Intel. Но если бы я что-то понимал в той ситуации, я бы сказал: «Почему бы нам не вернуть обратно того парня, кото- ▶



рый создал эту компанию, и понимает в ней все. Почему бы нам не вернуться назад и не нанять Стива, чтоб он вернулся и возглавил компанию?»

Оглядываясь назад, кажется настолько очевидным, что было бы правильно поступить так. Мы не сделали этого, поэтому я виню себя за эту ошибку. Это избавило бы Apple от того опыта нахождения в предсмертном состоянии, который они имели.

Одна из проблем, из-за которых меня уволили, заключалась в расколе внутри компании по вопросу о том, что нужно делать компании. Часть людей хотели, чтобы Apple была в большей степени компьютерной компанией. Они хотели открыть архитектуру и лицензировать ее. Другие люди, среди которых был я, хотели взять методологию Apple — впечатление пользователя и все в этом духе и перейти с ней к следующему поколению продуктов, таких как Newton.

Но Newton провалился. Это было новое направление. Оно кардинально отличалось от прежних продуктов. В результате меня уволили. Затем у них было еще два CEO, которые оба лицензировали технологии, но ... они закончили с промышленным дизайном. Они выпустили компьютеры, которые выглядели, как любые другие компьютеры, и они больше не заботились о рекламе и связях с общественностью. Они просто уничтожили все. Мы как

раз собирались стать инжиниринговой компанией, и они почти привели компанию к банкротству в ходе этого процесса.

Я на самом деле убежден, что если бы Стив не вернулся тогда, когда он вернулся, если бы они прождали еще шесть месяцев, Apple ушла бы в историю. Она бы прекратила существовать, полностью прекратила.

Что он сделал? Он вернул компанию обратно туда, откуда она пришла, как будто он никогда не покидал ее. Он прошел путь назад целиком.

Так что в мое время, на самом деле, все, что мы делали, соответствовало его философии его методологии дизайна.

К сожалению, я был не так хорош в этой философии, как он. Выбор времени решает в жизни все. Это просто было не то время, когда можно было создавать потребительские товары, и в NeXT он был не более удачлив, чем мы были в Apple, но он справлялся лучше, чем мы. Вот, что он действительно сделал лучше: он построил лучшую операционную систему следующего поколения, которая в конечном итоге была объединена с операционной системой Apple.

Вопрос: Я хочу спросить о героях Джобса. Вы говорите, Эдвин Лэнд был одним из его героев?

Скалли: Да, я помню, как Стив и я поехали на встречу с доктором Лэндом.

Доктор Лэнд был изгнан из Polaroid. У него была своя лаборатория на Чарльз-ривер в Кембридже. Это был увлекательный вечер, мы сидели в большом конференц-зале с пустым столом, доктор Лэнд и Стив оба смотрели на центр стола все время, пока разговаривали. Доктор Лэнд говорил: «Я мог видеть, какой должна быть камера Polaroid. Она была столь же реальна для меня, как если бы лежала передо мной, еще до того, как я сконструировал первый экземпляр».

И Стив ответил: «Да, именно таким образом я видел Macintosh». Он сказал, что если бы спросил кого-то, кто раньше использовал только персональный калькулятор, каким должен быть Macintosh, то не получил бы удовлетворительного ответа. Способа провести исследование потребительского спроса на него просто не существовало, так что ему пришлось пойти и создать Мас, а затем показать его людям и спросить, что они думают?

Они оба имели эту способность не изобретать, но открывать продукты. Оба сказали, что эти продукты существовали всегда — дело только в том, что никто никогда не видел их раньше. Мы были теми, кто открыл их. Камера Polaroid всегда существовала, и Macintosh всегда существовал — а потом их обнаружили. Стив был восхищен доктором Лэндом. Он был очарован этой поездкой.

Вопрос: О каких других героях он говорил?

Скалли: Он очень близко сошелся с Россом Перо.

Росс Перо посетил офис Apple несколько раз, и еще он посетил завод Macintosh. Росс имел системное мышление. Он создал компанию EDS (Electronic Data Systems) и был предпринимателем. Он верил в большие идеи - идеи, меняющие мир. Он был еще одним героем.

Акио Морита совершенно определенно был одним из его великих героев. Он был предпринимателем, который создал Sony и сделал это с помощью великолепных продуктов, а для Стива всегда были важны продукты. ■



Александр Люстик

советы и SEO-откровения

Автор: Елена Гутникова webmasters.ru

Саша привет, расскажи в целом, как за последние 3 месяца изменился Key Collector? Что добавилось, а что еще только в планах или на стадии реализации? Планируется ли выпустить версию под MacOS?

За последние 3 месяца проделано много работы, можно сказать, что Key Collector был переписан с точки зрения архитектуры – полностью. В первую очередь мы работали над стабильностью программы и переходом на базу данных, что позволило пользователям больше не беспокоиться о потере данных, т.к. они автоматически без всяких сохранений попадают в БД.

Разумеется, мы постоянно внедряем новый функционал в программу, будь то новые источники информации о ключевых словах или фишки для удобства работы с запросами, распределения их на различные группы и работа с каждой из них отдельно, в рамках одного проекта.

Также была увеличена скорость работы, и в ближайшие месяцы будет заново переписана система виртуализации данных, что значительно ускорит процесс обработки данных, а значит, нужно будет меньше системных ресурсов для работы программы. Конечно, все это касается больших проектов, которые нацелены на трафиковое продвижение, где количе-

ство запросов идет на сотни тысяч.

В планах развития внедрение новых функций, которые позволят сделать работу над ключевыми словами еще более удобной, еще более автоматизированной (хотя куда уж больше 😊).

Также в планах выход на зарубежный рынок, это значит только одно – расширение функций программы для англоязычных пользователей, что в свою очередь влечет подключение большего объема источников данных и более плотной работой с Google Adwords и другими западными системами.

При всем при этом, пользователи программы по-прежнему будут получать бесплатные обновления, а также более удобный сервис поддержки, над которым сейчас мы также работаем.

Спешу огорчить сектантов, увы, но версии отдельно под MAC не будет, как и версий под Linux. На всех этих OS есть возможность работы через виртуальную машину, да и количество пользователей на данных платформах, не велико. Пожалуй, их не более 5%, с учетом стоимости программы (а это всего 1070 рублей), поддержка версий под разные ОС не рассматривается, прежде всего, с точки зрения усилий по поддержке программы.

Планируются ли еще какие-либо программные продукты для автоматизации рутинных процессов SEO?

Да, конечно, новые продукты планируются к запуску, среди которых будут как бесплатные версии, так и платные с расширяемым (модульным) функционалом.

Затронута важная деталь в работе любого специалиста – автоматизация рутинных процессов, над этим мы и работаем очень плотно.

Мы хотим, чтобы оптимизаторы занимались своей прямой работой, а именно - анализом информации и развивали сайты, а не скатывались в рутину, которой в нашей работе хватает с избытком.

Но выпускать сырой продукт мы не хотим, поэтому релиз нового софта был отложен. Также мы понимаем, что без качественной документации выпускать на рынок софт не стоит, поэтому новые продукты сразу будут сопровождаться детальным мануалом и видео-уроками.

Мы приобрели огромный опыт при работе с Key Collector (к слову, продукт в коммерческой эксплуатации уже 1 год!). В данный момент мы знаем множество подводных камней, о которых ранее мы даже не задумывались, поэтому новый софт будет выпускаться с учетом всех требований пользова- ➤

телей, как к программе, так и к поддержке продукта.

Раскрывать детали, наверное, пока рано, всему свое время.

Помнится, ты писал пост о том, как используешь внутренние ресурсы сайта для увеличения поискового трафика. Есть ли мысли и это дело как-то автоматизировать, т.е. получить на выходе целевую страницу, тексты ссылок и страницы-доноры?

Да, конечно, этот процесс и так автоматизирован. Я думаю, в скором времени пользователи Коллектора обрадуются внедрению этого функционала в программу.

Было принято решение внедрить его именно в Key Collector, т.к. напрямую затрагивается область работы с ключевыми словами, распределение их по целевым страницам, в целях улучшения внутренней оптимизации сайта (спалил тему).

Ты ярый сторонник того, что «вечных» ссылок не бывает, а бывают ссылки с единоразовой оплатой за размещения. Какими ссылками пользуешься ты для продвижения клиентских и собственных проектов?

Я люблю называть вещи своими именами, мне не нравятся маркетинговые «заманухи», поэтому «вечное» понятие мной не принимается, как и «мнимые гарантии» любого вида услуг.

Для продвижения проектов, как своих, так и клиентских, используются разные ссылки, в зависимости от их задач. Разумеется, есть доля ссылок с единоразовой оплатой и они выбираются скрупулезно. Их доля увеличивается постепенно, причем это не биржевые варианты, а именно адресная работа с веб-мастерами, хотя биржевые тоже присутствуют.

Арендные ссылки (MainLink и Sape), разумеется, никуда не исчезли, для НЧ запросов по-прежнему нет ничего лучше, тем более нужна определенная масса, которую ссылками с единоразовой оплатой достигать не

целесообразно и довольно дорого.

Кроме того, намекаю – не ссылками едиными живут проекты...

Как ты относишься к ссылкам, размещаемым на новостных ресурсах, а также постовым с полноценных СДЛ-блогов?

Конечно, положительно, любой СДЛ блог или новостник может стать не просто источником SEO-ссылки, но и отличным PR-инструментом, если статью грамотно продвинуть в ТОП. Но это касается статей. Про вечные ссылки – негативно не отношусь. У всех ссылок есть свои задачи, если их разделять – то можно использовать любые ресурсы.

Появился Рейкьявик, ты что-то изменил глобально в своих стратегиях продвижения сайтов?

Я давно работаю с трафиком, поэтому Рейкьявик не внесет глобальных изменений в стратегию развития сайта. Подчеркну – развития сайта!

Кто до сих пор работает на несколько запросов в ТОП-е, тем будет сложнее, т.к. все более и более персонализированными будут результаты, в зависимости от того, что искал ранее пользователь. Это нормальная ветвь развития.

Сложнее будет анализировать выдачу, точнее, прогнозировать, но все это скорее в недалеком будущем, чем сейчас, ведь пока идет настройка, тестирование.

Если клиент приходит, мягко говоря, с плохим сайтом, ты предпочитаешь его не брать или взять с условием переработки сайтов?

Если клиент не хочет работать над сайтом вообще, то вне зависимости от того плохой или хороший сайт у него в наличии – я не берусь за такие сайты. Это не имеет никакого смысла в долгосрочной перспективе, а «быстро срубить денег» не та стратегия, по которой хочется развиваться дальше. Не хочу работать с сайтами, которые никому не нужны, даже за деньги.

Хотя такие клиенты редкость. Сейчас практически на любом сайте есть что переделывать, во благо бизнеса. И к счастью, клиенты это отлично понимают.

Только совместная работа над сайтом способна дать не просто положительный, а огромный эффект как с точки зрения SEO, так и с точки зрения конверсии.

Есть ли у вас собственные разработки в плане автоматизации покупки ссылок в Sape?

Были, есть, будут. Не только Sape. Автоматизация контроля ссылок – очень важный момент, он не зависит даже от уровня влияния ссылок, с точки зрения алгоритма поисковой системы.

Нужно знать, что покупается, что остается, что работает неправильно, что будет работать неправильно – вся эта информация нужна для анализа.

Пользуешься ли ссылочными агрегаторами для автоматизации или на текущий момент не настолько им доверяешь и предпочитаешь работать ручками?

Нет, ссылочными агрегаторами я не пользуюсь. Вопрос не в доверии, вопрос в другом. Если есть отточенная методика рабо-



ты со ссылками, то в принципе, единственная функция агрегатора – проверять эти ссылки после покупки.

Если есть автомат, который это делает, причем по большому количеству параметров, чем любой из существующих агрегаторов, то по сути – уже не нужно больше ничего.

Да и бюджеты на продвижение, реальные, в среднем ниже в раз 10, чем через прогнозная стоимость через агрегаторы.

Продвижение сайтов – это бизнес, причем желательно, чтобы он был ТВОЙ, а не чей-то, поэтому делегирование некоторых задач имеет смысл только в случае отсутствия автоматизации процессов на своей стороне.

На рынок постоянно выходят новые инструменты, так или иначе позволяющие автоматизировать те или иные процессы в работе оптимизатора. Каких сервисов на данный момент не хватает лично тебе?

Очень часто новые инструменты не адаптированы к потребностям специалистов, поэтому лучше сделать что-то свое, чем пользоваться чужим, не настроенным детально.

Именно поэтому запуск софта является интересной задачей, по сути – это переключивание разрозненных скриптов в единое целое, в один десктопный инструмент, который позволит заниматься любимым делом, а не рутинной.

Некоторые говорят, что рынок постепенно переходит в стагнацию. Возможно, есть и другие мнения. Как ты считаешь, что сейчас происходит в целом с рынком поискового продвижения и куда он будет двигаться в этом году, и куда отправится в начале следующего?

Рынок как всегда отправится на...й! Наверное, такой прогноз ожидается, исходя из вопроса?

Я не вижу никакой стагнации рынка. Если затрагивать отдельные сегменты, например, продажу ссылок, то, возможно, это направление стаг-

нирует, но рынок живет не только ссылками.

О какой стагнации рынка мы говорим, если рекламные бюджеты в сети идут вверх? Деньги никуда не уходят, они только приходят в данное направление, просто распределяются иначе, чем было 2-3 года назад.

Как ты считаешь, насколько сложнее стало «добывать» целевой поисковый трафик по сравнению с прошлым годом?

Смотря что считать мусором. Если рассматривать в общем и целом, то, наверное, трудней, т.к. с каждым годом растет конкуренция и специалисты все более и более дотошно отсеивают направления и запросы, по которым не стоит двигаться, а это означает только одно – за целевой трафик борется большее количество конкурентов.

С другой стороны, умение отсеивать мусор не приходит просто так, поэтому здесь есть еще поле для деятельности, многие до сих пор работают вслепую, просто с Wordstat'ом.

К великому сожалению, разгон мозга после долгого простоя, это не апгрейд компьютера (т.е. невозможно заменить одну деталь). Нужно развиваться целостно, постепенно.

Тебя и твои проекты, а также клиентов коснулись санкции за поведенческие факторы? Поделись опытом в этом направлении.

Нет, данные санкции не коснулись проектов, с которыми я работаю. Но теперь я могу точно сказать, что среди моих клиентов есть те, кто находится под санкциями.

Каким образом? Все довольно просто, из компаний, которых коснулись санкции, валом валит народ.

Теперь у меня есть подопечные, с которыми ведется работа по исправлению ситуации. Поделиться богатым опытом не могу – данная проблема слишком свежа, чтобы рассказывать о чем-либо со 100% вероятностью и точностью. Работаю в этом направлении, для анализа нужно больше времени, а прошло с момента санкций не так много.

Какие инструменты для анализа сайтов, для аудита, для съема позиций, для мониторинга тех же поведенческих факторов используете вы?

Есть, конечно, и свои наработки (о них не буду подробно, все равно еще рано), но в целом для большинства хватит обычной Яндекс.Метрики с Вебвизором, чтобы начать правильно работать со своей аудиторией.

Хотя дополнительно я использую сервис Woopra. У меня всегда включена эта программка на компьютере, позволяет очень интересные выводы делать, взаимодействовать с аудиторией постоянно и в режиме реального времени.

Практикуешь ли контекст для свежих клиентских сайтов и интересны твои наблюдения, есть ли корреляция между Директом и позициями в Яндексе?

Есть ситуация. Запустил контекст – позиции поднялись/опустились. Стоит ли делать вывод, что есть прямая корреляция между Директом и органической выдачей?

Кто-то скажет да, но сделает ошибку. Почему происходят движения в органической выдаче? – все просто, появляется аудитория на сайте и взаимодействует с этим сайтом, а значит, появляется и дополнительная информация у поисковой системы, которая решает, насколько релевантен запросу данный документ.

Что это? – поведенческие.

Практикую ли я нагон трафика из Директа для свежих клиентов? – да. А для старых клиентов? – тоже да.

Контекст позволяет на свежую страницу привести большой пласт аудитории для анализа, для того чтобы понять, насколько интересна информация для аудитории, а затем обработать данные и внести правки.

Это очень полезный пласт информации как для дальнейшего продвижения, так и для бизнес-целей.

Если говорить о заработке в интернет, то куда бы ты посоветовал молодежи направить свои усилия?

Все зависит от того, что считать «за-▶

работком». 1-100-1000\$-10000\$...? Для всех это разная сумма. Если новичком считать студента, который, как правило, не зарабатывает, то пожалуй, рекомендую направить свои усилия на изучение материалов, а после изучения – практические уроки по реализации полученных знаний.

Когда будет этот опыт, тогда можно будет выбрать и правильное направление. Что точно не нужно делать, так это расплыться на все подряд.

Умеете писать? – пишите. Умеете рисовать? – рисуйте. Умеете продвигать? – то не*** под новичков косить, пора давно зарабатывать деньги.

Суть не в том «куда», суть в том чтобы «направить» и «работать». Большинство выполнит «куда» и «направить», а через неделю забудет про «работать».

Куда - тут я не советчик, я давно работаю и проблема выбора не стоит, у каждого свои знания и умения, нужно именно их уметь задействовать.

Насколько менее безопасным сегодня стал заработок на ссылках по сравнению с началом года?

Вопрос только в прибыли и модели. Я сейчас почти не зарабатываю на ссылках, поэтому мне сложно анализировать ситуацию по данному направлению. Для разных сегментов ситуация разная, для ГС – сложнее, для СДЛ – проще.

Несколько советов и наблюдений - при помощи каких инструментов сегодня правильнее всего продвигать под русский Google?

Было, есть и будет всего 2 способа. Контент и ссылки. Что здесь изменилось за последние годы? На мой взгляд, ничего.

Инструмента 3: мозг, глаза и руки.

Как ты считаешь, приживутся и смогут ли развиваться биржи ссылок в англоязычном сегменте интернета?

Биржа – это инструмент автоматизации взаимодействий, поэтому не вижу причин не прижиться. Главное суметь свести аудиторию, покупающую и продающую, с поправкой на менталитет.

Насколько сложно в целом стало находить адекватных клиентов сегодня,

и вообще, есть ли тенденция к тому, что клиент стал разборчивее, придирчивее и грамотнее?

На самом деле, в данный момент очень легко стало работать с клиентами. В 90% случаев ко мне обращаются люди, знающие, чего хотят, хотя бы на уровне того, что видят происходящее с сайтом и хотят улучшить как конверсию, так и положение сайта в поисковых системах.

Это, несомненно, радует. Лет 5 назад приходилось объяснять, зачем это нужно, сейчас в лишние слова уже отпала необходимость. Есть прямое взаимодействие ради увеличения прибыли клиента, а значит, моей.

Конечно, заказчики стали более грамотными и требовательными, как к результатам, так и к сервису. Это очень хорошо, ведь это, несомненно, будет двигать рынок вперед.

Сколько процентов клиентов «сидят» на продвижении по трафику, а не на иных гарантиях?

В данный момент более 60% клиентов находятся на трафиковом продвижении с оплатой за целевой переход.

Еще около 30% находятся на трафиковом продвижении с работой по конверсии, здесь есть фикс платежей и неограниченное количество запросов, это наиболее выгодная схема для заказчиков.

Еще 10% - это старые клиенты, с которыми наше сотрудничество длится уже более 6 лет, у них формально в договоре продвижение по позициям, но по факту, я работаю с ними по максимально возможному целевому трафику с удержанием существующих позиций по важным запросам.

Лучшей гарантией для клиентов является работа с аудиторией сайта, с улучшением сайта, когда они видят увеличение конверсии по запросам и общее увеличение количества целевого трафика, а следовательно, и продаж.

Раз уж заговорили о гарантиях, открою небольшой секрет – в данный момент 70% клиентов находятся со мной в устных договоренностях, т.е. большинству вообще не нужны никакие бумажные «гарантии».

Потому что многие понимают, какой фикцией являются большинство до-

говоров по продвижению, а «мнимые» гарантии, которые на словах выглядят гарантиями, а в договоре превращаются в фиктивные оговорки - отсутствуют в договоре. Иными словами, никогда не подпишусь под гарантиями, которые

невозможно реализовать 😊

В целом начинается формироваться подход, что клиент платит определенный фикс, а ты работаешь над сайтом и добиваешься того, чтобы он приносил прибыль клиенту?

По факту – именно так. Это не потоковая схема продвижения, ведь проводится большая работа над сайтом, анализ трафика, увеличение конверсии. Все это требует большого количества времени, здесь не ограничиться шаблонной штамповкой (раньше можно было в одного держать, и 70 сайтов, сейчас такая схема просто не работает).

Клиенты это тоже понимают, любая работа должна быть оплачена, иначе это уже не работа.

Даже на трафиковом продвижении будет минимальный фиксированный платеж, ведь во многих случаях предстоит большая работа над сайтом, включающая в себя как структурную работу с программированием, так и тексты для сайта (коих бывает несколько тысяч).

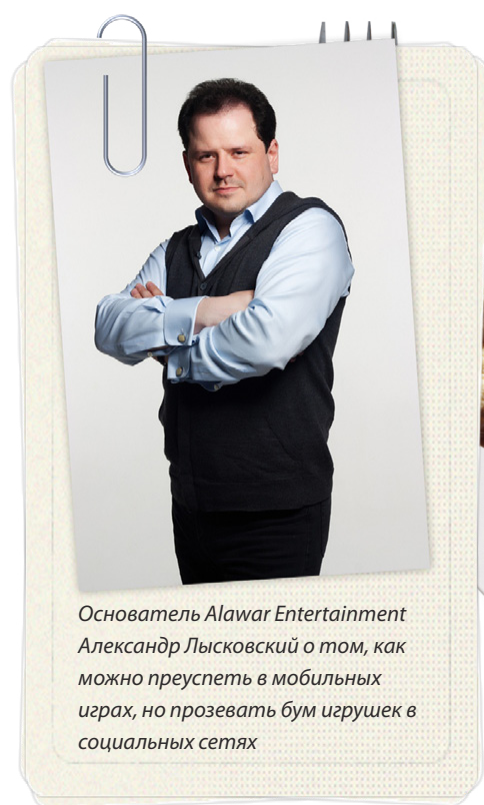
Ну и напоследок, твои практические советы тем, кто только начинает?

Для начала могу посоветовать начертить конечные цели, которых стремиться добиться.

Если вы хотите побаловаться с полгода/год и забросить – бросайте это дело сразу, не стоит тратить время, оно стоит дорого.

Если вы готовы потратить 2-3 года плодотворного в***вания, чтобы потом нормально зарабатывать и быть более свободным, уделять больше времени семье и саморазвитию, имея как активный, так и пассивный доход, то начинайте прямо сейчас.

Пошагово определяйтесь с тем, что действительно хотите делать! Начните, в конце концов хоть что-то делать, а не читайте кучи интервью, которые вам не принесут пользы, без желания работать! ■



«С играми так — если
создал хит, он тебя
кормит много лет»

Продолжаем публиковать интервью с известными российскими интернет-предпринимателями и специалистами. На этот раз о том, как устроен бизнес на играх и чем отличаются играющие на PC от Мас-геймеров, о секретах создания успешной игры и о том, почему успех игры зависит не от программиста, а также о том, в каком сегменте рынка разработчиков ждут небывалые прибыли, рассказывает основатель и генеральный директор компании Alawar Entertainment Александр Лысковский.

Максим Спиридонов: Мы собираемся поговорить об отрасли, которую иногда называют «серьезным рынком несерьезных игр». В фокусе нашего внимания — казуальные игры, как впрочем, и игры в целом. В гостях у нас основатель и генеральный директор компании Alawar Entertainment Александр Лысковский. Александр, казуальные игры, бывшие несколько лет назад очень сильно растущим сегментом, сегодня уступают играм для мобильных устройств, играм в социальных сетях. Что ждет их дальше? Полное вымирание?

Александр Лысковский: Раньше казуальные игры — игры для непрофильной аудитории — существовали только на персональных компьютерах, сейчас они присутствуют и на мобильных, и в соцсетях. Они теперь есть на всех платформах.

— И вы идете на все эти платформы?

— Да.

— Кстати, как лучше растолковать термин «казуальные игры» простым языком? ▶

— Такого термина нет, мы его сами придумали и периодически жалеем, что не назвали эти игры «алавариками», по аналогии с Xerox или Pampers. Существует слово casual, которое на Западе часто используется для обозначения неспециализированных, непрофессиональных игроков. Есть люди, которые считают себя игроками, и если остановить их на улице и спросить: «Кто ты?», то они скажут, что они игроки — играют в компьютерные игры. Обычно это подростки мужского пола от 15 до 25 лет. Они «рубятся» 30-40 часов в неделю, предпочитают шутеры, сложные стратегии, ММО. Они уходят в многочасовые рейды и готовы изучать свою игровую вселенную месяцами. Есть и другие. Если их остановить на улице и спросить: «Кто ты?», они никогда не скажут, что играют в игры. Они скажут, что работают менеджерами, воспитывают детей и так далее. Но у них всегда есть компьютер, смартфон, планшет или аккаунт в социальной сети. Они имеют доступ к различным играм на различных платформах и иногда в них играют. Это случайные игроки. Иногда они играют несколько часов в неделю. Случается, в обеденный перерыв, многие — в метро (на телефоне) или когда ждут чего-то.

— Сами по себе игры обычно достаточно простые по сюжету?

— Непрофессиональность игроков накладывает ограничение на сложность вхождения. Игрок должен очень быстро разобраться в игре. Он редко готов платить деньги сразу же, не видя игры, поэтому обычно используется shareware-модель. Пользователь бесплатно скачивает игру, играет полчаса-час. Если ему она нравится, то он за нее платит и получает доступ к полной версии. Либо он платит внутри игры за валюту или еще за что-то — это так называемая схема free to play. Начальное вхождение всегда короткое, потому что человек не готов тратить много времени и платить деньги.

— Ты сейчас, как и на момент создания компании, живешь в Новосибирске, хотя подавляющее большинство людей, оказавшись в такой ситуации, как ты, переезжает в Москву. Почему ты этого не сделал?

— Alawar — это не очень российская компания. Большая часть бизнеса, продаж и офисов находится не в России, поэтому перебираться в Москву смысла нет. Был бы смысл переезжать в Европу или Сиэтл.

— Сколько у вас офисов на сегодня?

— У нас есть люди в 10-11 странах.

— Где находится штаб-квартира?

— Главный офис — в Новосибирске. Это одна из причин, почему мне не хочется переезжать. Здесь работает слаженная команда из 150 человек. Они хорошо работают, и нет причины их увольнять, нанимать новых людей в Москве или заставлять всех покидать родных и близких и ехать в столицу.

— В каких еще городах и странах находятся ваши офисы?

— У нас шесть дочерних студий. Есть большой отдел в Барнауле. Мы присутствуем в Киеве, Днепрпетровске, Днепропетровске, Минске, в Америке, Польше и Турции.

— В Америке, Польше и Турции находятся разработка или продаж?

— Там сосредоточена продажа. Почти вся разработка находится в Восточной Европе, России, Белоруссии, на Украине.

— До Alawar вы делали игры, предназначенные для массового рынка?

— Никто их не покупал. Мы их делали для себя, друзья в них играли. Когда мы пытались из этого сделать коммерческий проект, выяснилось, что нужно делать игры другого качества, что программисты не главное, что нужны художники, музыканты, налаженные каналы продаж. Мы все это изучали, исследовали, и, когда мы разобрались в рынке, начался кризис 1998 года, и все проекты, которые к тому моменту были созданы, оказались никому не нужными. Резко изменилась ситуация в сфере продаж. Тогда уже были ком-

пьютеры, люди, которые умели что-то на них делать. Так появилась продажа игр через интернет на Западе. Тогда это называлось shareware. В 1998 году из-за кризиса появилась компания, которая создавала дешевые игры в Новосибирске и продавала их через сеть в США.

— Какие первые игрушки из числа успешных стоит упомянуть?

— Мы делали потрясающие клоны Расман. Нам казалось, что так можно делать. Как только на нас вышли правообладатели, мы прекратили продажи, но на вырученные деньги мы сумели создать команды, движки. Потом мы делали оригинальные игры, а не клоны.

— Что наиболее важно в бизнесе-успехе игры: удачно выбранный сюжет, рыночное попадание, хороший дизайн, оформление?

— Все важно. Очень важно знание рынка и попадание в аудиторию. Нельзя делать игру, которая никому не нужна. Необходимо понимать, кто покупает игру, где ее покупают, как доставить игры потребителям.

— Насколько детально вы моделируете образ своего покупателя? Это конкретные социально-демографические характеристики?

— Я не могу сказать, что это девочка Маша и мальчик Ваня, для которых мы делаем продукты, но мы хорошо знаем, для каких стран создаем продукты, какие каналы дистрибуции будем использовать и как там нужно преподносить игры. В казуальных играх важен этап вхождения. Человек скачал игру бесплатно, поиграл в нее, ему нужно быстро разобраться, как в нее играть. Учитывая, что он не является профессиональным игроком, он мог не видеть такие игры раньше. Очень важен начальный период, и мы представляем себе обычного пользователя обычного компьютера. Это «офисный планктон», домохозяйки и так далее.

— Правда ли, что основной пользователь казуальных игр — семейная женщина 30-35 лет?

— Если брать статистику PC-игр, то ▶

60-65% — женщины. Картина зависит от страны и сегмента. В мобильные игры играют немного другие люди. Те, кто играет на Mac, тоже отличаются от тех, кто играет на PC. Если мы говорим про продукты для PC, там другая аудитория. Усредняя, мы получаем абстрактную женщину.

— Чем отличаются от остальных владельцы Mac?

— Думаю, они больше привыкли к качеству интерфейсов и софта. На Mac лучше операционка, она меньше глючит и зависает. Выше требования к качеству. Понятно, что это люди творческих профессий: фотографы, художники, музыканты, студенты. Учеба у них часто связана с Mac. Apple активно продвигал свою продукцию, и, насколько я знаю, в американских школах стоят Mac.

— Несколько лет назад у вас однозначно доминировали игры под Windows. Это были программы, требующие оплаты. Сегодня casualность смещается в сторону мобильных устройств?

— В сторону мобильных устройств, социальных сетей, консолей. У нас вышло несколько социальных проектов, готовятся и другие. Несколько проектов вышло под консоли, PlayStation, но основное — это мобильные устройства (iPhone, iPad, Android).

— Так случайно получилось или это осознанный выбор?

— Это самый быстрорастущий рынок. Несмотря на то что все громко говорят о социальных играх, самый крупный и самый быстрорастущий сегмент игрового рынка — мобильные игры. В 2010 году он оценивался в \$8 млрд. Через пару лет будет \$15 млрд.

— Как сегодня выглядит структура Alawar?

— Главное, на чем мы специализируемся и в чем мы считаем себя успешными, — это поиск разработчиков, их поддержка в создании продуктов, финансирование, помощь в тестировании и локализации и продвижение

их по нашим собственным каналам и через дистрибьюторов. У нас есть продюсеры, сопродюсеры. Кроме того, мы подружился с некоторыми студиями, и они стали внутренними студиями Alawar, у нас есть совместные юридические лица. Alawar — это издательство примерно в 150 человек, 6 студий, где тоже суммарно человек 150, и несколько отделов продвижения в конкретных странах (Россия, Турция, Украина, Польша).

— Сегодняшняя структура эволюционным путем выростала из той фирмы, которая появилась в конце 1990-х?

— Нет, у нас было несколько важных внутренних революций. Мы начинали как обычные разработчики игр. Я игры разрабатывал, а мой партнер (из банковской сферы) умел заводить в Россию деньги из-за рубежа, принимать чеки в Новосибирске, обналачивать их и правильно платить с них налоги. Для 1999 года это была высшая акробатика. Ко всему прочему он хорошо знал английский, поэтому общался с западной прессой. С этого тандема все и началось. Постепенно мы научились делать игры и продавать их на Западе. В 2000 году к нам стали приходить другие разработчики игр и просили помочь им. Они делали игры, но не знали, как их продавать. Мы стали что-то им рассказывать, объяснять, подсказывать, брать на дистрибуцию их продукты. В итоге выяснилось, что у нас это так хорошо получается, что надо брать с них процент, а не просто помогать за кружкой пива. Мы стали заниматься такой деятельностью. Когда они говорили, что у них нет денег и что им надо доделывать игру, мы давали им деньги. Так мы и стали издательством. Через какое-то время мы закрыли собственный отдел разработки. Потом появилась и собственная разработка, но уже не на базе того, что было создано в 1999 году. Были сложившиеся коллективы, с которыми мы нашли точки соприкосновения. Они стали внутренними студиями.

— С твоей точки зрения как руководителя, что особенно хорошего есть у вас в компании?

— Так как компания постоянно раз-

вивается, мы 12 лет непрерывно растем. Мы называем себя «пространством возможностей». В Alawar всегда есть возможность поменять сферу деятельности, вырасти, заняться другими вещами — не теми, какими занимался раньше, переехать в другую страну. Это нравится людям. Многие сотрудники в менеджменте, топ-менеджменте начинали совсем не с нынешней своей профессии и не с нынешнего своего отдела, они блуждали внутри Alawar. Многим нравится, что можно прийти с идеей к генеральному директору или руководителям отделов и предложить ее, на что тебе скажут: «Давай! Делай!» — и из этого вырастет целое направление, в котором можно будет стать большим человеком. Например, тот парень, который у нас сейчас является главным мобильным продюсером (Саша Егшин), просто хотел делать игры для iPhone. Тогда iPhone еще не было, он только должен был появиться. Саша был Mac-фанатом и говорил: «Давайте придумаем игры!», на что ему сказали: «Давай! Делай!» Он сделал, и сейчас у нас выходит 40 проектов в год для iPhone. Он все это придумал с нуля и сам сделал.

— В 2007 году 23% акций у вас купил «Финам», верно?

— «Финам» в нас инвестировал. Они дали нам деньги на развитие.

— В 2010 году доля была выкуплена Almaz Capital Partners..

— Да. И мы, и «Финам» решили, что следующий шаг — зарубежный рынок. Нам нужен был партнер с западным опытом, с западной «пропиской», хорошо воспринимаемый в Долине. «Финам» таким фондом на тот момент не являлся. Мы решили, что правильным будет поменять фонд на «более западный». В качестве промежуточной стадии мы взяли российско-американский Almaz. Они прописаны в Америке и имеют огромный опыт работы в Кремниевой долине.

— До 2007 года вы полностью росли на свои деньги?

— Да, это проблема всего российского бизнеса тех времен: нигде нельзя ►

было найти займов и инвестиций. IT-бизнес не может ничего заложить в банке, чтобы ему дали кредит. Тогда было очень мало фондов, и все они готовы были разговаривать только с интеграторами, с поставщиками компьютеров.

— Сколько денег вы тогда получили, если не секрет?

— Конечно, секрет. За два с половиной года наличия у нас «Финама» они увеличили вложенные деньги, наверное, в 4,5 раза.

— Дальнейшие инвестиции планируются? Компании они нужны?

— Мы и сами неплохо растем. Мы прибыльная компания, у нас достаточно много свободных денег, которые мы активно реинвестируем. Все хотят предложить нам денег, но мы пока не очень понимаем для чего. Все, что можно делать, мы делаем сами. Как только появится возможность (видимо, она будет связана с какими-то слияниями), мы будем искать разные варианты: банковский кредит, IPO или инвестиции.

— В 2010 году вашу выручку оценивали в \$15-17 млн. Какая выручка у вас сейчас?

— Мы не раскрываем эти цифры. Сейчас это несколько десятков миллионов долларов. К тому моменту, когда компания будет готова к IPO, информация станет публичной, открытой, известной.

— Как строится взаимодействие с Almaz Capital Partners? Они активно участвуют в деятельности совета директоров? Или они «спящие» акционеры?

— Они участвуют в совете директоров. Они дали нам в совет директоров Павла Богданова и Сергея Белоусова. Опыт этих двух замечательных людей, по-разному смотрящих на одни и те же проблемы, нам очень нужен.

— Сегодня ваш главный рынок — Америка? Основные деньги приходят оттуда?

— 40% — это Америка, 35% — Россия.

— Каким образом вы структурируете деятельность с точки зрения

приоритетов? Если в Америке покупают больше, то вы вкладываете в маркетинг и продажи больше? Или ищете новые рынки? Какие у вас географические приоритеты?

— Современный IT давно не географический, а платформенный. Те же социальные сети не привязаны к географии. Facebook везде Facebook. Правильнее говорить об ориентации по платформам или сегментам рынка.

— У вас есть статистика по платформам? Видимо, все еще лидируют игрушки под Windows, на втором месте, наверное, социальные игры.

— В Alawar около трети оборота — мобильные игры. Я думаю, что через год на них будет приходиться половина оборота. Если года два-три назад мы были PC-компанией, которая создавала игры под персональный компьютер и пробовала новые платформы, то сейчас мы мультиплатформенные издатели. При создании игры мы сразу говорим, что есть десять платформ, на которых выйдет эта игра. Важнее платформы, а не язык или страна.

— Я знаю, что вы используете крупные площадки типа Yahoo! ►



Games, Big Fish Games. Это помогает распространять игры под Windows?

— В Америке около 20 крупных дистрибьюторов, они поделили весь рынок. Мы с точки зрения Америки не являемся крупным дистрибьютором, мы поставщики контента для этих дистрибьюторов. На других рынках иначе: в Восточной Европе мы лидируем, в Германии есть другие компании. Сделаю сравнение: можно самим продавать свои компьютеры, можно это делать через «Эльдорадо». Там есть такие же дистрибьюторы, как «Эльдорадо». Мы поставляем им продукцию, а они продают ее на своих площадках.

— Насколько различаются сценарии создания игры под Windows, мобильные устройства, социальные сети?

— Социальные сети — это другое. Все, что не касается социальности и большого количества пользователей в игре, — более или менее одинаковый процесс. Есть сценарий, продюсер, команда разработчиков, финансирование, тестеры, локализаторы, создается продукт, и либо потом он переходит на другие платформы, либо сразу пишется под несколько платформ. Если мы говорим о многопользовательских играх, все по-другому: там очень важно взаимодействие пользователей, важна серверная составляющая и нет явного использования наших стандартных игр, мы просто используем бренды. Мы сделали бренд «Веселая ферма», и при создании социальной игры на эту тему мы задействуем этот бренд, сама же игра создается с нуля.

— Какие деньги уходят на создание игр для разных платформ?

— Для PC — от \$50 000 до \$150 000. Если мы говорим о мобильных устройствах, то от \$10 000 до \$50 000.

— А для социальных сетей?

— В социальных сетях создание игры — это не главное. После создания игра не отдается в продажу, она постоянно обновляется, изменяется. Те деньги, за которые игра создана, иногда менее важны, чем те, которые

будут затем потрачены на эту игру. Нужен хостинг, маркетинг.

— Как различается срок жизни игр на разных платформах?

— На PC — года два-три, хотя у нас есть продаваемые с 2005 года игры, продажи которых несильно падают. Например, хит 2005 года Magic Ball. Уже много лет он продается на \$10 000 — 15 000 в месяц. Таких игр несколько.

— При этом они ничего не требуют? Их не нужно «кормить»?

— Да, с казуальными играми именно так. Если создал хит, то он тебя много лет кормит. С социальными сетями так не получится, эти игры нужно постоянно обновлять, улучшать. О сроке жизни мобильных игр говорить сложно, потому что мобильные игры сейчас совсем другие. Несколько лет назад под мобильными играми понимали нечто другое. Сегодня мы говорим о смартфонах, которые появились 2-3 года назад. Есть игры, созданные для iPhone 2-3 года назад, и они до сих пор продаются. Может, будут и через 3 года. Игра выпускается, и на первые 2-3 месяца приходится основной всплеск продаж, потому что идет активная реклама, раскрутка, потом — «хвост». Длина этого «хвоста» может быть внушительной, и количество денег в нем бывает не меньше, чем на всплеске.

— Вы последовательно придерживаетесь политики казуальных игр. Вы не думали заняться MMO и создать что-то наподобие World of Warcraft, Lineage?

— Конечно, это успешные, прибыльные и быстроразвивающиеся компании, у которых все хорошо, но почему мы должны идти в эту сферу? Сейчас основной тренд — мобильные игры. Мы в нем. Мы выпускаем несколько десятков мобильных проектов под все основные платформы. У нас на днях стартует проект с Windows Phone 7. Это первая казуальная игра, которая выйдет под маркой Xbox на Windows Phone 7. Издает ее Microsoft. Мы в тренде мобильного развития. Зачем нам нужны соседние ниши, мне непонятно. Там ненамного больше денег, а

в некоторых даже меньше. Например, в социальных сетях пока меньше денег. Ажиотаж есть, но каждое новое слово в IT-индустрии вызывает такой эффект. Мобильникам много лет, поэтому ажиотажа нет.

— Какое место в масштабах мира занимает Alawar? Вы крупные?

— Я думаю, что мы самая крупная компания на этом рынке за пределами США. Мы хотим, чтобы мы стали такой же крупной компанией на рынке мобильных игр. С социальными сложнее, потому что есть огромное количество компаний, которые специализируются именно на социальных играх. Для нас это способ привлечь аудиторию к нашим проектам.

— Вы собираетесь «входить в телевизор»?

— Мы думаем об этом. У нас есть несколько проектов с Samsung и другими телевизионными игроками. Если Apple выпустит телевизор, то это будет круто, потому что они встроит внутрь какой-то аналог iOS, добавят поддержку интернета и можно будет прямо с телевизора скачивать что-то с Appstore. Это будет сильная заявка.

— Ходят разговоры, что Apple выпустит телевизор?

— Как обычно, в их случае никто ничего точно не знает, но разговоры ходят. Все крупные игроки (Apple, Microsoft и Google) думают о том, чтобы зарабатывать деньги не только на железке, но и на контенте, а телевизор — это очень крупное потребление контента, связанное с фильмами, передачами и сериалами. Почему бы их тоже не продавать и не давать людям интерактивность на уровне новостей, игр, голосований? Мы очень спокойно относимся к тому, что будут появляться все новые и новые устройства, все новые и новые форматы развлечений. Если это хороший, «вкусный» рынок, мы туда обязательно приходим, что-то пробуем и думаем, нужно ли создавать еще один отдел в компании или это просто эксперимент. Например, были эксперименты с игровыми автоматами. Мы попробовали, но не получилось. ►

— Мог бы ты вспомнить свою самую большую неудачу и самый большой успех?

— Что касается лично меня, то, поскольку я занимаюсь стратегией, иногда я предпринимаю не очень верные действия, связанные с ней. Мы пытались выйти на шведский рынок и локализовали кучу игр на шведский, наняли там менеджера, и все это провалилось. Такие вещи я периодически делаю, и они стоят дорого для компании, и мне периодически их припоминают на заседаниях совета директоров.

— А какие наиболее заметные успехи? Скажем, уровня Angry Birds. Компанию сейчас оценивают в \$1,2 млрд. Они за месяц зарабатывали десятки миллионов евро при первичных вложениях €100 000.

— Из последних заметных вещей — это первые места на мобильных платформах. Примерно год назад у нас вышла игра Hotel Mogul. К тому моменту iPad продавался всего пару месяцев. Это была первая игра в топе, и она оставалась второй среди всех платных приложений довольно долго. Это был громкий успех для новой, быстрорастущей платформы. Потом были первые места у нашей игры «Сокровища Монтесумы». Это номер два в США, и это заметный успех. Понятно, что Angry Birds висел в тот момент на первом месте, но и второе место нас мотивировало и очень радовало. На разных платформах первые места у нас были, и мы очень этим гордимся. Это позволяет нам развивать линейку и общаться с владельцами платформ или социальных сетей. После истории с Hotel Mogul к нам пришел Microsoft и сказал: «Ребята, вы сделали мегасуперприложение для iPad, сделайте такое же для Windows Phone 7». Они оказывают нам большую маркетинговую поддержку, и они хотят, чтобы это было очень хорошее и громкое приложение. То же самое сейчас с Sony PSP.

— Мог бы ты нарисовать будущее казуальных игр, ниши рынка, в которой существует компания? Что изменится в этой сфере за 3-5 лет?

— Мы по-прежнему будем помогать разработчикам со всего мира создавать

хорошие игры. Мы будем находить этих ребят, финансировать их, продюсировать, прорабатывать с ними проекты и помогать им доносить игры до всех пользователей на всех устройствах. Я не считаю, что это расширение сферы деятельности. Мы, как и раньше, занимаемся хорошими играми для массовой аудитории, просто количество устройств возросло. У нас есть отдел локализации, он берет английскую игру, переводит ее на русский или немецкий язык, и точно так же мы сейчас относимся к платформам. Была игра для PC, и отдельные люди портируют ее на Android. Самих игр больше не станет. Мы сейчас выпускаем около 40 проектов в год. Мы считаем, что это много. Возможно, их надо сокращать. Но у нас будут громкие и заметные проекты.

— Это будущее компании. А будущее самого рынка?

— Будущее самого рынка заключается в том, что пользователь через какое-то время перестанет разбираться в платформах, железках и операционках. Он будет брать какое-то устройство (телефон, компьютер, телевизор или микроволновку), и на нем почти всегда будут какие-то интерактивные развлечения, в том числе игры, с цифровой доставкой. Не нужно будет что-то куда-то вставлять, надо будет зайти в меню, нажать кнопку и увидеть список доступных приложений, и там обязательно окажутся игры. Дальше — скачай эту игру, поиграй в нее и оплати, если она тебе нравится, либо не плати, но посмотри рекламу. Это будет всегда и на всех устройствах. А мы просто будем делать качественные игры.

— Несколько вопросов от слушателей. «Каким образом Alawar прозевал бум игрушек в социальных сетях? Почему, если утрировать, не стал огромным и страшным, как Zynga?»

— Мы придерживаемся стратегии медленного спуска с горы и завоевываем те рынки, которые уже сформировались, а не бежим туда сразу же. Рынок игр для социальных сетей молодой и быстрорастущий. Сейчас мы выпускаем несколько проектов и серьезно на него выходим. Пару лет назад мы не считали

это необходимым, потому что рынки, на которых мы работали, были крупнее и заметнее.

— «Когда молодой команде удачнее выбрать момент для поиска продюсера?»

— Сразу же. Как только появилась мысль о создании игры, общайтесь с большим количеством издателей и продюсеров, потому что хуже всего исправляются ошибки, совершенные на ранней стадии. Например, при придумывании геймплея, при выборе платформы. Это как сейчас делать игры под DOS. Многие разработчики совершают такие ошибки.

— «Как правильно искать, найти и выбрать дистрибьютора?»

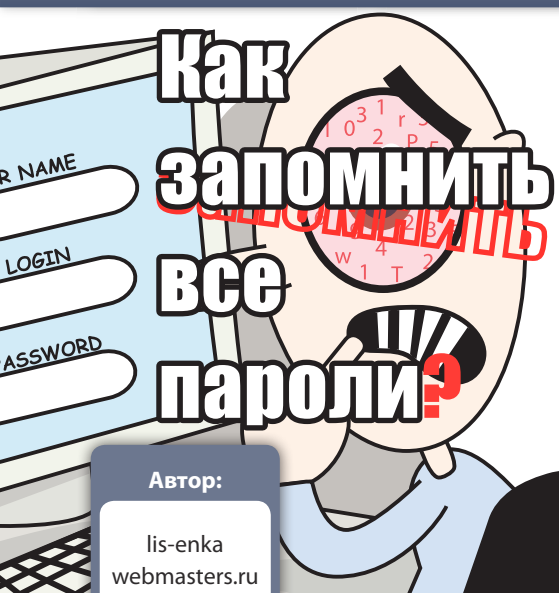
— Надо спрашивать, какие игры он уже дистрибутировал, каких успехов добился, потом спрашивать у разработчиков этих игр, как им работало с этим дистрибьютором. Мы не скрываем своих разработчиков. Мы имеем дело примерно с 40 командами, из них шесть — внутренние, остальные — внешние. С каждой командой можно связаться и спросить, как им работает с нами.

— «Когда и как пора понять, что идея никому не нужна, кроме самих разработчиков, и приступить к поиску и реализации новой идеи?»

— Чем раньше вы покажете продукт потенциальному пользователю или продюсеру, тем лучше. Прототипирование, показ, «коридорное тестирование» нужно проводить как можно раньше.

— Что такое «коридорное тестирование»?

— Это когда в коридоре вылавливается человек, ему показывается продукт и задается вопрос: «Как тебе?» Мы для таких вещей используем бухгалтерию. Продюсер заходит в бухгалтерию, отгрызает несчастных девушек от заполнения бесконечных бумажек, показывает им игру и спрашивает: «Вам понятно, что тут нужно делать?» Они начинают тыкать пальцем, и сразу становится ясно, что они не находят нужной кнопки. Он говорит им спасибо и уходит. ■



Пароли можно записать в один карманный компьютер, которым не смогут воспользоваться злоумышленники даже в случае утери. Осталось только создать такое устройство

Едва ли найдутся еще какие-то меры безопасности, создающие так много проблем, как эта старая вредная цепочка битов, известная под названием «пароль». Эксперты призывают делать их как можно длиннее и сложнее, никогда не использовать один и тот же пароль (несмотря на то что наши онлайн-аккаунты множатся) и хранить их только в голове. Эти советы можно обобщить, как это недавно [сделал в Twitter](#) исследователь систем безопасности Микко Хиппонен: «Выберите что-нибудь, что не можете запомнить, и не записывайте».

Кэмбриджский исследователь [Фрэнк Стаджано](#), занимающийся компьютерными технологиями, считает, что у него есть решение. На конференции Usenix он представил доклад об устройстве Pico — крошечном компьютере, который можно носить с собой и который выполняет роль аутентификатора для тысяч различных сервисов и устройств. Пользователям Pico больше не придется запоминать пароли, к тому же они теоретически будут защищены от фишинговых атак, угрозы выбора слабых паролей и даже от возможности кражи пароля подглядывающим из-за плеча

шпионом.

«Пользователь имеет надежное устройство под названием Pico, которое работает как своего рода протез памяти и освобождает от бремени запоминания аутентификационных комбинаций, превращая их из «того, что вы знаете» в «то, что вы имеете», — пишет Стаджано.

К несчастью, Pico пока не существует. Но воображаемая система, которую Стаджано описывает на [двух десятках страниц](#), включает множество умных идей о том, как создать такой универсальный «брелок» для ключей.

В видении Стаджано Pico должен быть маленьким вычислительным устройством со встроенным радиопередатчиком и камерой. Оно будет использовать [криптографию с открытым ключом](#) для создания и хранения тысяч криптографических пар (общедоступный и личный ключи), по одному для каждого приложения или гаджета, которые нужно открывать пользователю. Когда прибор уже знаком с этими программами и оборудованием, камера Pico читает визуальный код на экране регистрации или на устройстве, чтобы его узнать, а потом с помощью радиосигнала отправляет зашифрованное сообщение на удаленный сервер регистрации — чтобы только сервер мог расшифровать его с помощью индивидуального секретного ключа.

Сервер в свою очередь отправляет зашифрованное сообщение, которое может расшифровать только уникальный компьютер Pico с помощью собственного секретного ключа, приписанного данному сервису.

Два главных преимущества этой технологии — помимо того что прибор может генерировать идеальные сложные пароли, которые пользователь не обязан запоминать — заключаются в том, что таким образом подтверждается идентичность не только пользователя, но и сервиса или устройства, которым он пытается воспользоваться. Например, нельзя будет создать ложную страницу регистрации Gmail и украсть пароль пользователя. И логиры клавиатуры — программы, перехватывающие пароль в том момент, пока его набирают на клавиатуре — устареют: благодаря волшебству криптографии открытого ключа будет невозможно перехватить аутентификационные

сообщения и вычислить оба личных ключа. Криптография открытого ключа подразумевает, что ключи шифровки отличны от ключей дешифровки, которые к тому же никогда не показываются.

А как быть с очевидной проблемой утери этого универсального устройства? Вот здесь схема Стаджано становится особенно интересной. Он предлагает пользователям вдобавок к Pico носить с собой «братьев» Picosiblings — другие устройства, которые занимают только тем, что постоянно отправляют радиосигналы короткого радиуса действия, чтобы Pico знал, что они рядом. Эти Picosiblings, по мысли Стаджано, должны быть объектами, которые пользователь носит постоянно — например, очки, ремень, кошелек, украшения — даже пирсинг, парики, зубные протезы и подкожные имплантаты.

Благодаря этим «братьям» Pico понимает, когда его теряют, и тут же выключается. (Пользователь может восстановить информацию с резервных копий, которые, как уточняет Стаджано, будет автоматически делать зарядное устройство к Pico.)

На случай, если Pico украли вместе со всеми «братьями» (например, если ограбят шкафчик для переодевания в бассейне), прибор будет иметь биометрическую проверку — например, устройство для считывания отпечатков пальцев, которое нужно будет использовать примерно раз в день.

Самой трудной частью превращения этой фантастической схемы Стаджано в реальность будет, вне сомнений, переделка всех приложений и гаджетов для стандартного использования Pico. Но некоторые приложения уже понемногу движутся в этом направлении. И Google, и Facebook сейчас позволяют пользователям подписываться на «двухфакторную аутентификацию», при которой используется также приложение на смартфоне или текстовое сообщение, чтобы удостовериться, что регистрирующийся пользователь имеет при себе определенный телефон.

Pico можно интегрировать и в смартфон, пишет Стаджано. Но его схема, конечно, пойдет гораздо дальше, чем двухфакторная аутентификация Google и Facebook. Чтобы пройти регистрацию, будет важен всего один фактор — обладание самим Pico. С одним «но» — если кто-нибудь его создаст. ■

Автор:

Роман Веснин
habrahabr.ru

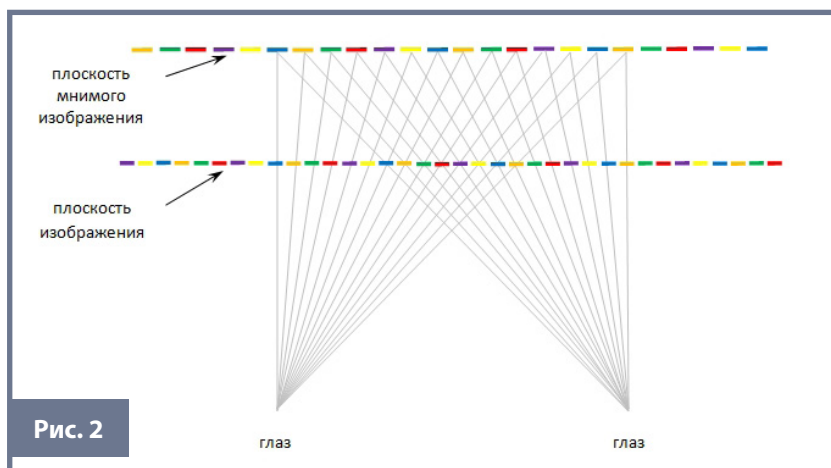
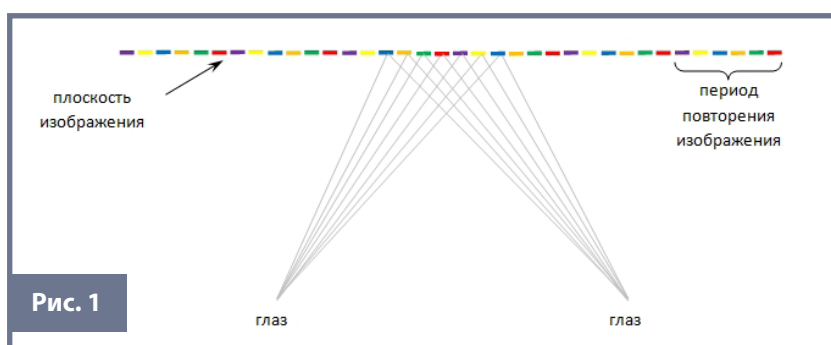
Стереои́зображение — это просто

В данной статье я хочу рассказать, как можно самостоятельно создать стереои́зображение при помощи графического редактора и небольшой программы.

Введение

Для начала рассмотрим, как устроено стереои́зображение и как на него смотреть. В первую очередь оно состоит из повторяющихся фрагментов (рис. 1).

Очень важно, чтобы ширина повторяющегося фрагмента была меньше, чем расстояние между глаз. Для комфортного просмотра стереои́зображения ширина должна колебаться в пределах от $\frac{1}{3}$ до $\frac{2}{3}$ данной величины. Чем ширина больше, тем более глубокое (объемное) изображение можно получить. Чем меньше ширина, тем меньше устают глаза от просмотра стереои́зображения. ▶



Чтобы увидеть объемное изображение, необходимо сфокусировать взгляд за плоскостью рисунка таким образом, чтобы изображения для левого и правого глаз совпали. Для изображения, целиком состоящего из повторяющихся фрагментов, вы будете наблюдать точно такое же мнимое изображение (ровная плоскость). Единственное отличие – вам будет казаться, что рисунок стал располагаться дальше (рис. 2).

Как получается объемное изображение?

Чтобы изображение стало объемным, некоторые его части должны восприниматься, как более близкие, другие – как более удаленные. Данного эффекта можно достичь путем сдвига элементов изображения.

На рисунке 3 показано, как изменилось мнимое изображение, после сдвига двух элементов (красного и фиолетового) реального изображения на одну позицию влево (рис. 3).

На мнимом изображении образовалось два артефакта одинакового размера, равного размеру сдвинутых элементов. Артефакт, находящийся слева от места сдвига, воспринимается, как расположенный более близко к наблюдателю (перед плоскостью мнимого изображения). Артефакт справа, наоборот, воспринимается как расположенный за плоскостью мнимого изображения. О положении элементов мнимого изображения, которые одновременно видны только одному глазу (отмечены пунктирной линией), мозг «додумывает» самостоятельно.

Замечу, что сдвиг не должен превышать половины ширины повторяющейся части. Когда размер сдвига близок к этой границе, глазу трудно определить, выпукла или вдавнена наблюдаемая область.

Если мы хотим изображать объекты, расположенные только перед плоскостью мнимого изображения, нам необходимо уметь бороться с нежелательными артефактами. Для этого достаточно осуществить сдвиг элементов в каждом периоде, расположенном ►

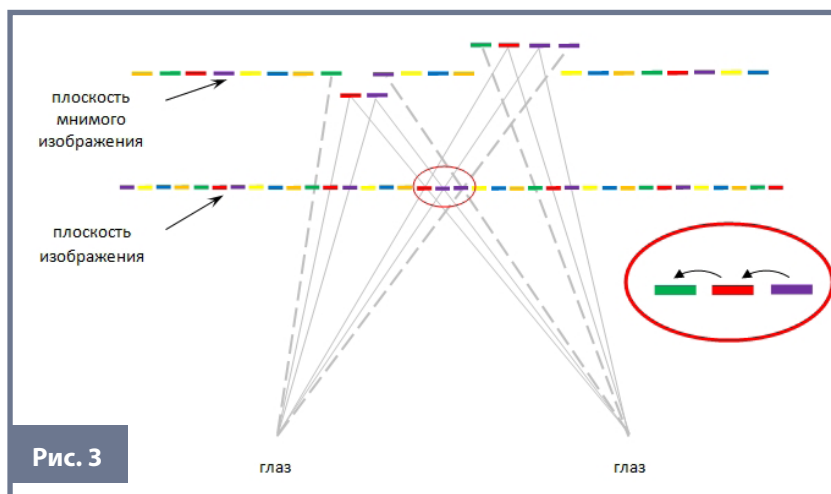


Рис. 3

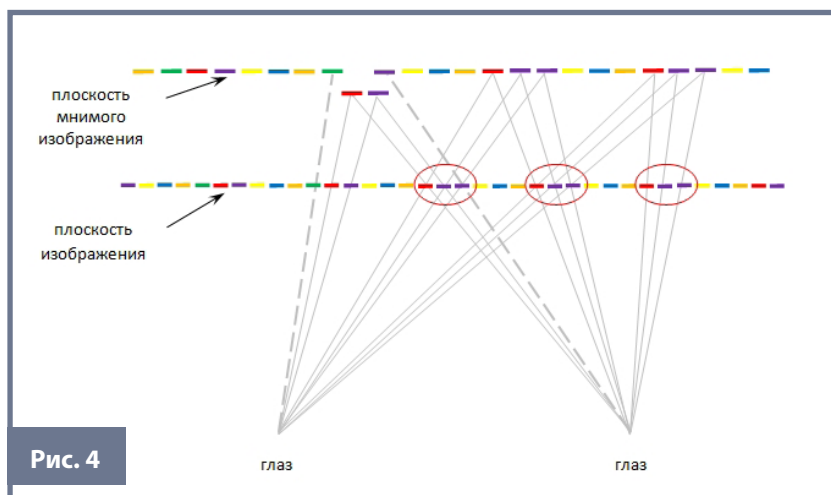


Рис. 4

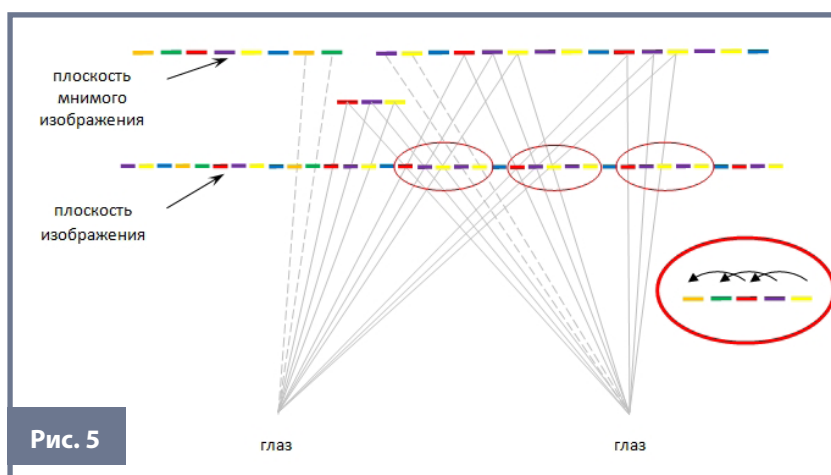


Рис. 5

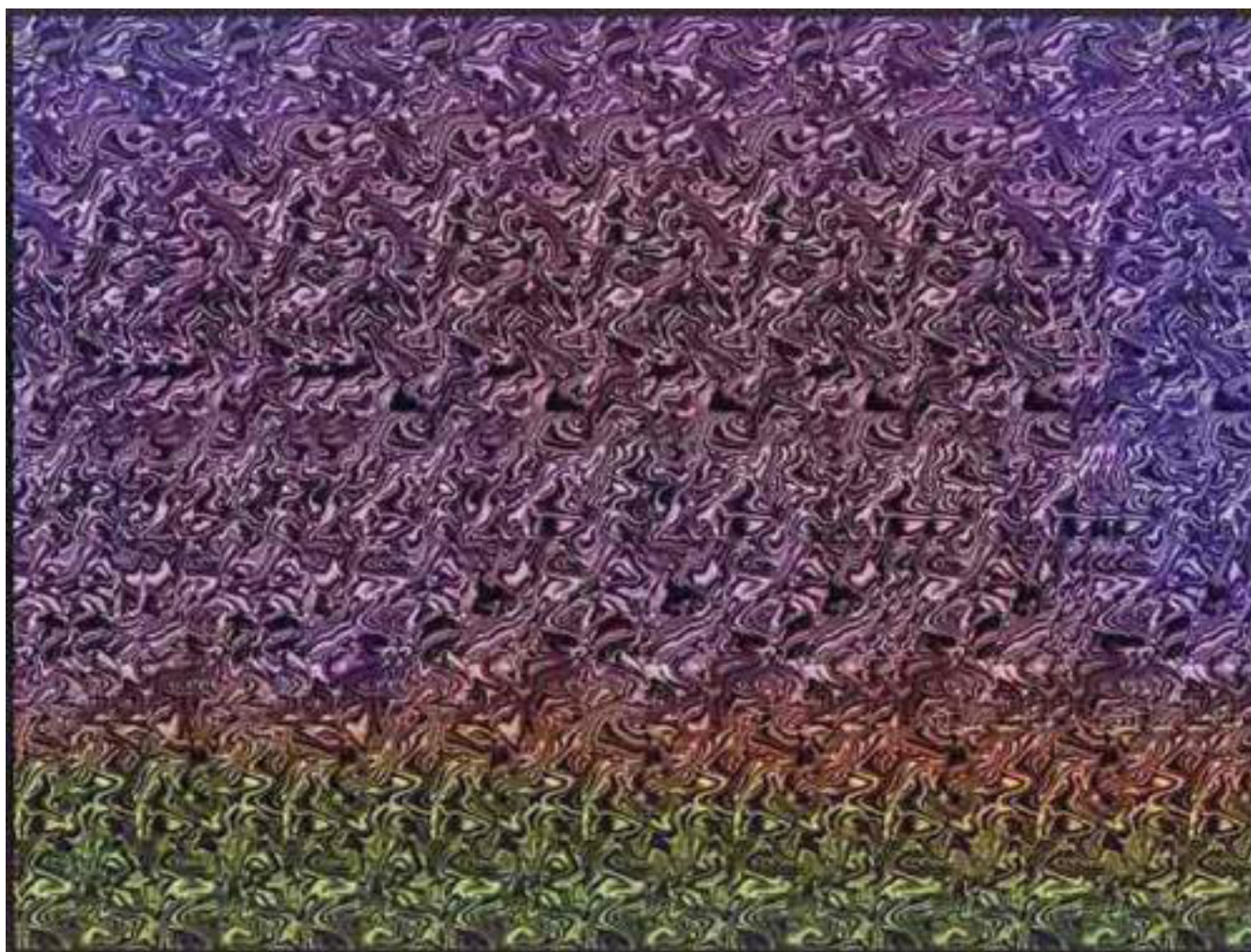
справа от первого места сдвига (рис. 4).

Осталось показать, как на мнимом изображении формируются объекты с разной степенью удаленности. Для наглядного примера на сей раз осуществим сдвиг трех элементов влево, но не на одну позицию, а на две. Сравнивая рисунки 4 и 5, видим, что на последнем рисунке за счет сдвига на большее количество позиций, мы получили объект, расположенный ближе, чем при сдвиге на одну позицию (рис. 5).

```
private Bitmap GenerateStereopicture(Bitmap
bitmapMask)
{
    // Переводим маску в массив сдвигов
    int w = bitmapMask.Width;
    int h = bitmapMask.Height;
    int[][] mask = new int[w][];
    for (int x = 0; x < w; x++)
    {
        mask[x] = new int[h];
        for (int y = 0; y < h; y++)
```

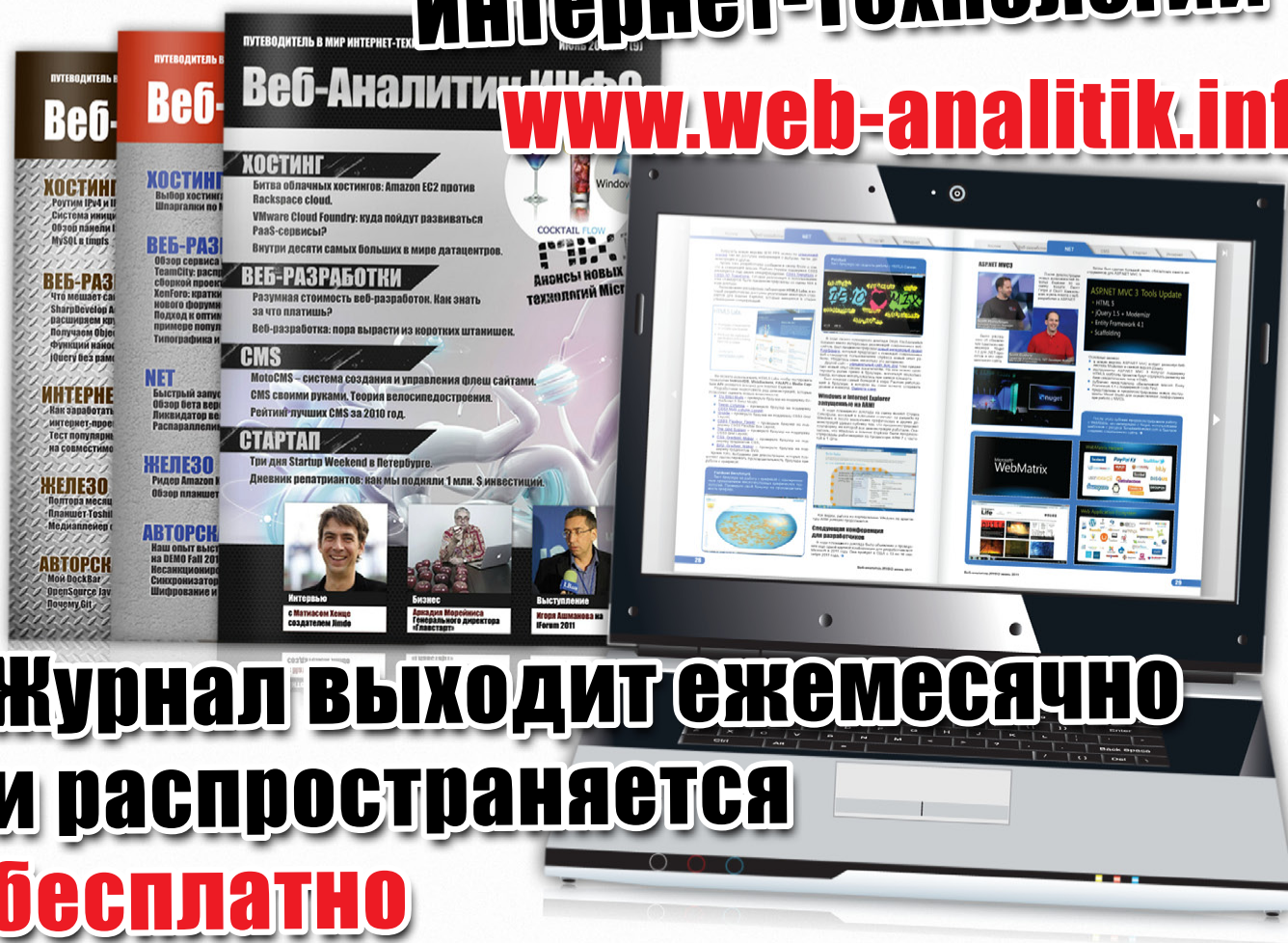
```
        mask[x][y] = bitmapMask.GetPixel(x, y).R
/ 32;
    }
    // Создаем фон
    int s = 100;
    Bitmap stereoImg = GetNewStereopicture(w +
s, h, s);
    // Сдвигаем каждый пиксел
    for (int y = 0; y < h; y++)
        for (int x = 0; x < w; x++)
            if (mask[x][y] > 0)
            {
                Color pixel = stereoImg.GetPixel(x +
mask[x][y], y);
                for (int i = x + s; i < w + s; i += s)
                    stereoImg.SetPixel(i, y, pixel);
            }
    return stereoImg;
}
```

■



Путеводитель в мир Интернет-Технологий

www.web-analitik.info



**Журнал выходит ежемесячно
и распространяется
бесплатно**

