Программирование на Java

Методическое руководство для преподавателей

27 апреля 2003 года

Авторы документа:

Николай Вязовик (Центр Sun технологий МФТИ) <vyazovick@itc.mipt.ru>
Евгений Жилин (Центр Sun технологий МФТИ) < gene@itc.mipt.ru>

Аннотация

Данное руководство предназначено для преподавателей информатики в ВУЗах России, принимающих участие по постановке курсов преподавания Java в рамках программы поддержки процесса обучения информационным технологиям в ВУЗах, представленной представительством компании Sun Microsystems в странах СНГ и Московским физико-техническим институтом (МФТИ) .

Методика преподавания опирается на четырехлетний опыт преподавания курсов по Java студентам МФТИ. По инициативе Sun Microsystems курс переработан таким образом, чтобы максимально облегчить процесс сертификации по Java для студентов, прослушавших курс. При подготовкие курсов использован опыт создания информационных систем на основе Java технологий, накопленный Центром Sun технологий МФТИ и Центром открытых систем и высоких технологий МФТИ.

Предлагается переподготовка преподавателей для чтения курсов по Java на основе совместного учебного центра Sun Microsystems и Центра дополнительного профессионального образования МФТИ.

Оглавление

Лекция 1. Что такое Java? История создания.

Содержание лекции.

1. Что такое Java?

Что знают о Java обычные пользователи персональных компьютеров и Интернета? Что говорят о нем разработчики, которые не занимаются этой технологией профессионально?

Java широко известна как новейший объектно-ориентированный язык, легкий в изучении и позволяющий создавать программы, которые могут исполняться на любой платформе без каких-либо доработок (кросс-платформенность). Еще с Java почему-то всегда связана тема кофе (изображения логотипов, названия продуктов и т.д.). Программисты могут добавить к этому описанию, что язык похож на упрощенный C или C++ с добавлением garbage collector'a - автоматического сборщика "мусора" (механизм освобождения памяти, которая больше не используется программой). Также известно, что Java ориентирована на Интернет, и самое ее распространенное применение - небольшие программы, называющиеся апплеты, которые запускаются в браузере и являются частью HTML-страниц.

Критики в свою очередь утверждают, что язык вовсе не так прост в применении, многие замечательные свойства лишь обещаются, а на самом деле не очень-то работают, а

главное - программы на Java исполняются ужасно медленно. Следовательно, это просто некая модная технология, которая только на время привлечет к себе внимание, а затем исчезнет, как и многие другие.

Однако некоторые факты не позволяют согласиться с такой оценкой. Во-первых, со времени официального объявления Java прошло около семи с половиной лет - многовато для "просто модной технологии". Во-вторых, конференция разработчиков JavaOne, которая была впервые организована в 1996 году, уже через год собрала более 10.000 участников и стала крупнейшей конференцией по созданию программного обеспечения в мире (каждый следующий год число участников росло примерно на 5.000). Специальная программа Sun, объединяющая разработчиков Java по всему миру, Java Developer Connection также была запущена в 1996 году, через год она насчитывала более 100.000 разработчиков, а в 2000 году - более 1.5 миллионов. На данный момент число программистов на Java оценивается в 3 миллиона.

Было выпущено 5 основных версий языка, начиная с 1.0 в 1995 году и заканчивая 1.4 в феврале 2002 года. Следующая версия 1.5 планируется на 2003 год. Все версии и документация к ним всегда были доступны для бесплатного получения на официальном веб-сайте Java http://java.sun.com/. Один из первых продуктов для Java - JDK 1.1 (средство разработки на Java) в течение первых трех недель после объявления был загружен более 220.000 раз. Последняя версия 1.4 была загружена более 2 миллионов раз за первые 5 месяцев. Практически все ведущие производители программного обеспечения лицензировали технологию Java и регулярно объявляют о выходе новых продуктов, построенных на ней. Это и "голубой гигант" IBM, и создатель платформы Macintosh фирма Apple, и лидер в области реляционных БД Oracle, и даже первейший конкурент фирмы Sun - корпорация Microsoft - лицензировала Java еще в марте 1996 года.

В следующей главе описывается краткая история зарождения и развития идей, приведших к появлению Java, что поможет понять, чем на самом деле является эта технология, каковы ее истинные свойства и отличительные качества, для чего она предназначена, и откуда взялось такое количество различных слухов и мнений.

2. История создания Java

Если поискать в Интернете историю создания Java, то можно выяснить, что изначально язык назывался Oak (дуб), а работа по его созданию началась еще в 1990 году с довольно скандальной истории внутри корпорации Sun. Эти факты верны, однако на самом деле все было еще интереснее.

2.1. Сложности внутри Sun Microsystems

Действительно, события начинают разворачиваться в декабре 1990 года, когда бурного развития WWW (World Wide Web - "всемирная паутина") никто не мог еще даже предсказать. Тогда компьютерная индустрия была поглощена взлетом персональных компьютеров. Увы, фирма Sun Microsystems, занимающая заметную долю рынка серверов и высокопроизводительных станций, по мнению многих сотрудников и внешних экспертов не могла предложить ничего интересного для обычного пользователя "персоналок" - для них компьютеры от Sun представлялись "слишком сложными, очень некрасивыми и чересчур "тупыми" устройствами" [3].

Поэтому Скотт МакНили (Scott McNealy), член совета директоров, президент и CEO (исполнительный директор) корпорации Sun, не был удивлен, когда 25-летний хорошо зарекомендовавший себя программист Патрик Нотон (Patrick Naughton), проработав всего 3 года, объявил о своем желании перейти в компанию NeXT. Они были друзьями, и Патрик объяснил свое решение просто и коротко - "они все делают правильно". Скотт задумался на секунду, и произнес историческую фразу. Он попросил Патрика перед уходом описать, что, по его мнению, Sun делает не верно. Надо было не просто рассказать о проблеме, но предложить решение, не оглядываясь на существующие правила и традиции, как будто в его распоряжении имеются неограниченные ресурсы и возможности.

Патрик Нотон выполнил просьбу, вложив в свое письмо все свои мысли и сердце. Он безжалостно раскритиковал новую программную архитектуру NeWS, над которой фирма работала в то время, а также привел свои восторженные оценки только что объявленной операционной системы NeXTstep. Среди его предложений были: привлечь профессиональных художников-дизайнеров, чтобы сделать пользовательские интерфейсы Sun приятными и привлекательными; выбрать одно средство разработки и сконцентрировать усилия на одной оконной технологии, а не на нескольких сразу (Нотон был вынужден поддерживать сотни различных комбинаций технологий, платформ и интерфейсов, используемых в компании); наконец, уволить практически каждого из Window Systems Group (они будут просто не нужны, если выполнить предыдущие условия).

Конечно, Нотон был практически уверен, что его письмо просто проигнорируют, но все же задержал свой переход в NeXT в ожидании какой-нибудь ответной реакции. Однако она превзошла все ожидания.

МакНили разослал это письмо всему управляющему составу корпорации, а те переслали его своим ведущим специалистам. Откликнулись буквально все, и, по общему мнению, Нотон описал то, что все подозревали, но боялись признать. Решающей оказалась поддержка Билла Джоя (Bill Joy) и Джеймса Гослинга (James Gosling). Билл Джой - один из основателей и вице-президент Sun, а также участник проекта по созданию операционной системы UNIX в университете Беркли. Джеймс Гослинг пришел в Sun в 1984 году (до этого он работал в исследовательской лаборатории IBM) и был ведущим разработчиком, в частности автором первой реализации текстового редактора EMACS на C. Эти люди имели огромный авторитет в корпорации.

Чтобы развить этот успех, надо было предложить какой-то совершенно новый, новаторский проект. Нотон объединился с группой технических специалистов, и они просидели до 4.30 утра, обсуждая базовые концепции такого проекта. Их получилось всего три: главное - потребитель, и все строится исключительно в соответствии с его интересами; небольшая команда должна спроектировать небольшую же аппаратно-программную платформу; и воплотить эту платформу в устройстве, которое будет предназначено для персонального пользования, удобно и понятно в обращении - компьютер для обычных людей.

Этих идей оказалось достаточно, чтобы Джон Гейдж (John Gage), руководитель научных исследований Sun, смог организовать презентацию для высшего руководства корпорации. Нотон изложил свои условия, которые он считал необходимыми для успешного развития этого предприятия: команда должна расположиться вне офиса Sun, чтобы не испытывать никакого сопротивления революционным идеям; проект будет секретным для всех, кроме высшего руководства Sun; аппаратная и программная платформы могут быть не совместимыми с любыми продуктами Sun; на первый год группе необходим миллион долларов.

2.2. Проект Green

5 декабря 1990 года, в день, когда Нотон должен был перейти в компанию NeXT, Sun сделала ему встречное предложение. Руководство согласилось поддержать все его пожелания. Ожидаемый результат - "создать что-нибудь необычайное". 1 февраля 1991 года Патрик Нотон, Джеймс Гослинг и Майк Шеридан (Mike Sheridan) вплотную приступили к проекту, который получил название Green.

Цель они выбрали себе амбициозную - выяснить, что станет следующей волной развития компьютерной индустрии (первыми считаются появление полупроводников и персональных компьютеров), и что необходимо разработать для успешного участия в ней. С самого начала проект не рассматривался как чисто исследовательский, задача была создать реальный продукт, устройство.

Во время общения на ежегодном собрании Sun весной 1991 года, Гослинг заметил, что компьютерные чипы получили необычайное распространение, они применяются в видеомагнитофонах, тостерах, даже в дверных ручках гостиницы, где они жили! Тем не менее, до сих пор в каждом доме можно увидеть до трех пультов дистанционного управления - для телевизора, видеомагнитофона и музыкального центра. Так родилась идея разработать небольшое устройство с жидкокристаллическим сенсорным экраном, которое бы общалось с пользователем через анимацию, показывая, чем можно с его помощью управлять и как. Чтобы создать такой прибор Нотон начинает работать над специализированной графической системой, Гослинг берется за программное обеспечение, а Шеридан занимается бизнес-вопросами.

В апреле 1991 года команда выезжает из офиса Sun в новое помещение и отключается даже от внутренней сети корпорации. Они закупают разнообразные бытовые электронные устройства, такие как игровые приставки Nintendo, телевизионные приставки, пульты дистанционного управления, и играют в многочисленные игры целыми днями, чтобы лучше понять, как сделать пользовательский интерфейс легким в понимании и использовании. В качестве идеального примера Гослинг отмечал, что современные тостеры с микропроцессорами имеют точно такой же интерфейс, что и тостер его мамы, который служит уже 42 года. Он считал, что к этому должны стремиться все бытовые устройства, как, например, современный сигнал цветного телевидения можно принять с помощью черно-белого телевизора 50-х годов производства.

Очень быстро исследователи обнаружили, что практически все устройства построены на самых разных центральных процессорах. Это означает, что добавление новых функциональных возможностей крайне затруднено, так как необходимо учитывать ограничения и, как правило, довольно скудные возможности используемых чипов. Когда же Гослинг побывал на музыкальном концерте, где смог воочию наблюдать сложное переплетение проводов, огромное количество колонок и полуавтоматических прожекторов, которые, казалось, согласовано танцуют в такт музыке, он понял, что будущее за объединением сетей, компьютеров, и других электронных устройств в единую согласованную инфраструктуру.

Сначала Гослинг попытался модифицировать C++, чтобы создать язык для написания программ, минимально ориентированных под конкретные платформы. Однако очень скоро стало понятно, что это практически невозможно. Основное достоинство C++ - скорость программ, но отнюдь не их надежность. А надежность работы для обычных пользователей должна быть так же абсолютно гарантирована, как и совместимость обычных электрических вилки и розетки. Поэтому в июне 1991 года Гослинг, который написал свой первый язык

программирования в 14 лет, начинает разработку замены C++. Создавая новую директорию и раздумывая, как ее назвать, он выглянул в окно, и взгляд его остановился на растущем под ним дереве. Так язык получил свое первое название - Oak (дуб). Спустя несколько лет, на основе маркетинговых исследований имя сменили на Java.

Всего несколько месяцев потребовалось, чтобы довести разработку до стадии, когда стало возможным совместить новый язык с графической системой, над которой работал Нотон. Уже в августе команда смогла запустить первые программы, демонстрирующие работу будущего устройства.

Само устройство, по замыслу создателей, должно было быть размером с обычный пульт дистанционного управления, работать от батареек, иметь привлекательный и забавный графический интерфейс, и в конце концов стать любимой (и полезной!) домашней игрушкой. Чтобы построить этот не имеющий аналогов прибор, находчивые разработчики применили "технологию молотка". Они попросту находили какой-нибудь аппарат, в котором были подходящие детали или микросхемы, разбивали его молотком и таким образом добывали необходимые части. Так были получены основной жидкокристаллический экран, сенсорный экран и миниатюрные встроенные колонки. Центральный процессор и материнская плата были специально разработаны на основе высокопроизводительной рабочей станции Sun. Было придумано и оригинальное название - *7, или Star7 (с помощью этой комбинации кнопок можно было ответить с любого аппарата в офисе на звонок любой другого телефона, а поскольку редко кого из них можно было застать на рабочем месте, эти слова очень часто громко кричались на весь офис). Для придания привлекательности интерфейсу был создан забавный персонаж Дьюк (Duke), который всегда был готов помочь пользователю в выполнении его задач. В дальнейшем он стал неразлучным спутником Java, счастливым талисманом, он присутствует во многих документах, статьях, примерах кода.

Задача была совершенно новая, не на что было опереться, не было никакого опыта, никаких предварительных наработок. Команда трудилась, не прерываясь ни на один день. В августе 1991 года произошла первая демонстрация для Билла Джоя и Скотта МакНили. В ноябре группа снова подключились к сети Sun по модемной линии. Чем дальше развивался проект, тем больше новых членов присоединялось к команде разработчиков. Примерно в то время было придумано название для той идеологии, которую они создавали, - "1st Person" (условно можно перевести как "первое лицо").

Наконец, 4 сентября 1992 года Star7 был завершен и продемонстрирован МакНили. Это было небольшое устройство с 5" цветным (16 бит) сенсорным экраном без единой кнопки. Чтобы включить его, надо было просто дотронуться до экрана. Весь интерфейс был построен как мультик - никаких меню! Дьюк перемещался по комнатам забавно нарисованного, "мультяшного" дома, чтобы управлять им, надо было просто водить пальцем - никаких специальных органов управления. Можно было взять виртуальную телепрограмму с нарисованного дивана, выбрать передачу и "перетащить" ее на изображение видеомагнитофона, чтобы запрограммировать его на запись.

Результат превзошел все ожидания! Стоит напомнить, что устройства типа карманных компьютеров (PDA), начиная с Newton, появились заметно позже, не говоря уже о цветном экране. Это было время 286i и 386i процессоров Intel (486i уже появились, но были очень дороги) и MS DOS, даже мышь еще не была обязательным атрибутом персонального компьютера.

Руководство Sun было просто в восторге, появилось отличное оружие против таких могучих конкурентов, как HP, IBM и Microsoft. Созданная технология была способна отнюдь не

только демонстрировать мультики. Объектно-ориентированный язык Oak был готов стать достаточно мощным инструментом для написания программ, которые могут работать в сетевом окружении. Его объекты, свободно распространяющиеся по сети, работали бы на любом устройстве, начиная с персонального компьютера и заканчивая обычными бытовыми видеомагнитофонами и тостерами. На презентациях Нотон рисовал область применения Oak, изображая домашние компьютеры, машины, телефоны, телевизоры, банки и соединяя их единой сетью. Целое приложение, например, для работы с электронной почтой, могло быть построено в виде группы таких объектов, причем им было не обязательно располагаться на одном устройстве. Более того, как язык, ориентированный на распределенную архитектуру, Oak имел механизмы безопасности, шифрования, процедур аутентификации, причем все эти возможности были встроенные и, таким образом, незаметные и удобные для пользователя.

2.3. Компания FirstPerson

Крупные компании-производители, такие как Mitsubishi Electric, France Telecom, Dolby Labs, заинтересовались новой технологией, начались переговоры. Шеридан подготавливает бизнес-план с оригинальным названием "Beyond the Green Door" ("За зеленой дверью"), в котором предлагает Sun учредить дочернюю компанию для продвижения платформы Oak на рынок. 1 ноября 1992 года создается компания FirstPerson, которую возглавила Вэйн Роузинг (Wayne Rosing), перешедшая из Sun Labs. Арендуется роскошный офис, число сотрудников возрастает с 14 до 60 человек.

Однако в дальнейшем оказалось, что стоимость подобного решения (процессор, память, экран) составляет не менее $50. Производители же бытовой техники привыкли платить ничтожные суммы за дополнительную функциональность, облегчающую использование их продуктов.

В это время внимание компьютерной индустрии захватывает идея интерактивного телевидения, создается ощущение, что именно оно станет следующим революционным прорывом. Поэтому, когда в марте 1993 года Time Warner объявляет конкурс для производителей компьютерных приставок к телевизору для развертывания пробной сети интерактивного телевидения, FirstPerson полностью переключается на эту задачу. И снова неудача - победителем оказывается Джеймс Кларк (James Clark), основатель Silicon Graphics Inc., несмотря на то, что технологически его предложение уступает по возможности Oak. Впрочем, через год проект Time Warner и SGI проваливается, а Джеймс Кларк создает компанию Netscape, которая еще сыграет важную роль в успехе Java.

Другим потенциальным клиентом стал производитель игровых приставок 3DO. Понадобилось всего 10 дней, чтобы портировать Oak на эту платформу, однако после трехмесячных переговоров, директор 3DO потребовал полные права на новый продукт, и сделка не состоялась.

Наконец, в начале 1994 года стало понятно, что интерактивное телевидение оказалось ошибкой. Было много ожиданий, но им не суждено стать реальностью. Анализ состояния FirstPerson показал, что компания не имеет ни одного клиента или партнера, и ее дальнейшие перспективы довольно туманны. Руководство Sun требует немедленного составления нового бизнес-плана, позволяющего компании начать приносить прибыль.

2.4. World Wide Web

В погоне за призраком интерактивного телевидения многие участники компьютерного рынка совершенно пропустили поистине эпохальное событие. В апреле 1993 года Марк Андриссен (Marc Andreessen) и Эрик Бина (Eric Bina), работающие в Национальном Центре Суперкомпьютерных Приложений (National Center for Supercomputing Applications, NCSA) при университете Иллинойс, выпустили первую версию графического браузера ("обозревателя") Mosaic 1.0 для WWW. Хотя Internet существовал на тот момент уже около 20 лет, имеющимися протоколами связи (FTP, telnet и др.) пользоваться было очень неудобно, и Глобальная Сеть использовалась лишь в академической и государственной среде. Mosaic же основывался на новом языке разметки гипертекстовых документов (HyperText Markup Language, HTML), который с 1991 года разрабатывался в Европейском Институте Физики Частиц (CERN) специально для представления информации в Интернете. Этот формат позволял просматривать текст и изображения, а главное - поддерживал ссылки, с помощью которых можно было одним нажатием мыши перейти как на другую часть той же страницы, так и на страницу, которая могла располагаться совсем в другой части сети и географического мира. Именно такие перекрестные обращения, используя которые пользователь мог совершенно незаметно для себя посетить множество узлов Интернета, и позволили считать все HTML документы связанными частями единого целого - Всемирной Паутины (World Wide Web, WWW).

И самое важное - все эти новые достижения были совершенно бесплатно доступны для всех желающих. Впервые обычные пользователи персональных компьютеров безо всякой специальной подготовки могли пользоваться глобальной сетью не только для решения рабочих вопросов, а для поиска информации на самые разные темы. Количество документов в пространстве WWW стало расти экспоненциально, и очень скоро сеть Интернет стала поистине Всемирной. Правда, со временем обнаружилось, что такой способ организации и хранения информации очень напоминает свалку, в которой крайне трудно найти данные по какому-нибудь конкретному вопросу, однако, эта тема относится к совершенно другому этапу развития компьютерного мира.

Итак, совершенно необъяснимым способом Sun не замечает зарождения новой эпохи. Технический директор Sun впервые увидел Mosaic лишь три месяца спустя! И это притом, что около 50% серверов и рабочих станций в сети Интернет были произведены именно Sun.

Новый бизнес-план FirstPerson ставил цель, которая была неким промежуточным шагом от интерактивного телевидения к возможностям Интернета. Идея заключалась в создании платформы для кабельных компаний, конечными пользователями которой были бы обычные пользователи персональных компьютеров, объединенные сетями таких компаний. Используя технологию Oak, разработчики могли бы создавать приложения, по функциональности аналогичные CD-ROM программам, однако обладающие интерактивностью, позволяющей пользователям легко обмениваться любой информацией через сеть. Ожидалось, что такие сети в итоге и разовьются в полноценное интерактивное телевидение, и тогда Oak станет полноценным решением для этой индустрии. Об Интернете и Mosaic пока не говорилось ни слова.

По многим причинам этот план не устроил руководство Sun (он плохо соответствовал главному ожиданию - новая разработка должна была привести к увеличению спроса на продукты Sun). Из-за отсутствия перспектив половина сотрудников FirstPerson была переведена в только что созданную команду Sun Interactive, которая продолжила заниматься

мультимедиа-сервисами уже без Oak. Все предприятие оказалось под угрозой бесславной кончины, однако в этот момент Билл Джой снова оказал поддержку проекту, который вскоре дал миру платформу Java.

Когда создатели FirstPerson наконец обратили внимание на Интернет, они поняли, что функциональность тех сетевых приложений, для которых они создавали Oak, очень близки к WWW. Билл Джой вспомнил, как он двадцать лет назад принял участие в разработке UNIX в Беркли, и затем эта операционная система получила широчайшее распространение благодаря тому, что ее можно было загрузить по сети совершенно бесплатно. Такой принцип бесплатного распространения коммерческих продуктов создал саму WWW, тем же образом компания Netscape вскоре стала лидером рынка браузеров, так многие технологии получили возможность захватить долю рынка в кратчайшие сроки. Эти новые идеи при поддержке Джоя окончательно убедили руководство Sun, что Интернет может стать воскрешением платформы Oak (кстати, этот новый проект поначалу называли "Liveoak"). В итоге Джой садится писать очередной бизнес-план и отправляет Гослинга и Нотона начинать работу по адаптации Oak для Интернета. Гослинг пересматривает программный код платформы, а Нотон берется за написание "убойного" приложения, которое бы сразу продемонстрировало всю мощь Oak для Интернета.

В самом деле, эти технологии прекрасно подошли друг другу. Языки программирования всегда играли важную роль в развитии компьютерных технологий. Мейнфреймы не были особенно полезны, пока не появился Cobol. Благодаря языку Fortran от IBM, компьютеры стали широко применяться для научных вычислений и исследований. Basic - самый первый продукт от Microsoft - позволил всем программистам-любителям легко создавать программы для своих персональных компьютеров. Язык C++ стал основой для развития графических пользовательских интерфейсов, таких как Mac OS и Windows. Создатели Oak сделали все, чтобы эта технология сыграла такую же роль в программировании для Интернет.

Несмотря на то, что к середине 1994 года WWW достиг невиданных размеров (конечно, по меркам того времени), веб-страницы продолжали быть больше похожими на обычные бумажные издания, чем на интерактивные приложения. По большей части вся работа в сети заключалась в отправке запроса на веб-сервер и получении ответа, который содержал обычный статический HTML-файл, отображаемый браузером на стороне клиента. Уже тогда функциональность веб-серверов расширялась с помощью CGI (Common Gateway Interface). Эта технология позволяла по запросу клиента запускать обычную программу на сервере и ее результат отсылать обратно в качестве ответа. Поскольку в то время скорость каналов связи была невысокой (хотя, похоже, пользователи никогда не будут удовлетворены возможностями аппаратуры), то клиент мог ждать несколько минут, чтобы лишь увидеть сообщение, что он ошибся в одной букве запроса. Динамическое построение графиков при таком способе реализации означало бы генерацию GIF-файлов в реальном времени. А ведь зачастую клиентские машины являются полноценными персональными компьютерами, которые могли бы брать значительную часть работы взаимодействия с пользователем на себя, разгружая сервера.

Вообще, клиент-серверная архитектура, просто необходимая для большинства сложных корпоративных (enterprise) приложений, обладает рядом существенных технических сложностей. Основная идея - разместить общие данные на сервере, чтобы создать единое информационное пространство для работы многих пользователей, а программы, отображающие и позволяющие удобно редактировать эти данные, выполняются на клиентских машинах. Очень часто в корпорации используется несколько аппаратных платформ (это может быть как "историческое наследие", так и следствие того, что различные

подразделения, решая разные задачи, нуждаются в различных компьютерах). Следовательно, приложение необходимо развивать сразу в нескольких вариантах, что существенно удорожает поддержку. Кроме того, обновление клиентской части означает, что нужно перенастроить все компьютеры компании в кратчайший срок. А ведь часто обновлениями могут заниматься несколько групп разработчиков.

Попытка придать интернет-браузерам возможности полноценного клиентского приложения встречает еще большие трудности. Во-первых, обычные сложности предельно возрастают - в Интернете представлены поистине все существующие платформы, а количество и географическая распределенность пользователей делает быстрое обновление просто невозможным. Во-вторых, особенно остро встает вопрос безопасности. Через сеть удивительно быстро распространяется не только важная информация, но и вирусы. Текстовая информация и изображения не несут в себе никакой угрозы для клиентской машины, другое дело - исполняемый код. Наконец, приложения с красивым и удобным графическим интерфейсом, как правило, имели немаленький размер, недаром основным способом их распространения являлись CD-ROM'ы. Понятно, что для Интернета необходимо было серьезно поработать над компактностью кода.

Если оглянуться на историю развития Oak, то становится понятно, что эта платформа удивительным образом отвечает всем перечисленным требованиям интернет-программирования, хотя и создавалась во времена, когда про WWW никто даже и не думал. Видимо, это говорит о том, насколько верно предугадали развитие индустрии участники проекта Green.

2.5. Возрождение Oak

Для победного нашествия Oak не хватало последнего штриха - браузера, который бы поддерживал эту технологию. Именно он должен был стать тем самым "убойным" приложением Нотона, которое завершало почти пятилетнюю подготовительную работу перед официальным объявлением новой платформы.

Браузер назвали WebRunner. Нотону потребовался всего один выходной, чтобы написать основную часть программы. Это было в июле, а в сентябре 1994 года WebRunner уже демонстрировался руководству Sun. Небольшие программы, написанные на Oak для распространения через Интернет, назвали апплетами (applets), и на первом примере такого апплета Дьюк махал ручкой своим создателям.

Следующая демонстрация происходила на конференции, где встречались разработчики интернет-приложений и представители индустрии развлечений. Когда Гослинг начал презентацию WebRunner, аудитория не проявила большого интереса, решив, что это просто клон Mosaic. Тогда Гослинг провел мышкой над сложной трехмерной моделью химической молекулы.

Следуя за курсором, модель поворачивалась по всем направлениям! Сейчас это, возможно, не производит такого впечатления, однако надо представить, что в то время это было подобно переходу от картинки к кинематографу. Следующий пример демонстрировал анимированную сортировку. Вначале изображался набор отрезков разной длины. Затем синяя и красная линии начинали бегать по этому набору, сортируя их по размеру. Пример тоже нехитрый, однако наглядно демонстрирующий, что на стороне клиента появляется полноценная программная платформа. Оба эти апплета сейчас являются стандартными примерами и входят в состав Java Development Kit любой версии. Успех этой демонстрации,

которая закончилась бурными аплодисментами, показал, что Oak и WebRunner готовы устроить революцию в Интернете, так как все участники конференции начали переосмысление возможностей, которые предоставляет Всемирная Сеть.

Кстати, к тому времени в начале 1995 года, когда стало ясно, что официальное объявление уже близко, за дело взялись маркетологи. По результатам их исследований Oak был переименован в Java, а WebRunner стал называться HotJava. Многие тогда сильно удивлялись, что дало повод для такого решения. Основная легенда гласит, что Java - это сорт кофе (такой кофе действительно есть), который очень любили программисты. Видимо, примерно по той же логике появилось и название HotJava (горячая Java). Тема кофе навсегда останется в названиях (технология создания компонент названа Java Beans - зерна кофе, специальный формат для архивирования файлов с Java программами JAR - банка с кофе и т.д.) и логотипах, а сам язык критики стали называть "для кофеварок". Впрочем, сейчас все уже привыкли и не задумываются над названием, возможно, это и было целью (а тем, кто продолжает высказывать недовольство, приводят альтернативные варианты, которые рассматривались - Neon, Lyric, Pepper или Silk).

Согласно плану спецификация Java, реализация платформы и HotJava должны были свободно распространяться через Интернет. С одной стороны, это позволяло в кратчайшие сроки распространить технологию по всему миру и сделать ее стандартом де-факто для интернет-программирования. С другой стороны, при помощи всего сообщества разработчиков, которые бы высказывали свои замечания, можно было гораздо быстрее устранить все возможные ошибки и недоработки. Однако в конце 1994 года пока лишь считанные копии были распространены за пределы Sun. В феврале 1995 года выходит, возможно, первый пресс-релиз, сообщающий, что вскоре альфа-версии Oak и WebRunner будут доступны для всеобщего внимания.

Когда это случилось, команда стала считать каждый случай, когда кто-то загрузил продукты для просмотра. Вскоре пришлось считать уже сотнями. Затем решили, что если удастся достигнуть 10.000, то это будет означать "просто ошеломляющий успех". Дожидаться 10.000 пришлось совсем не так много времени, как они могли предполагать. Интерес нарастал лавинообразно, после просмотров приходило большое количество писем, и мощности интернет-канала стало все время не хватать. На письма всегда отвечали очень подробно, что по началу можно было делать, не отрываясь от работы. Затем по очереди стали назначать одного разработчика, чтобы он в течение недели только писал ответы. Наконец, потребовался специальный человек, так как письма приходили уже по 2-3 тысячи в день.

Вскоре руководство Sun осознало, что такой ошеломляющий успех Java не имеет никакого бюджета или плана для рекламы и других акций продвижения на рынок. Первым таким событием становится публикация 23 марта 1995 года в газете Sun Jose Mercury News статьи с описанием новой технологии, в которой приводился адрес официального сайта http://java.sun.com/, который по сей день является основным источником информации по Java.

2.6. Java выходит в свет

Наконец, вся подготовительная работа стала подходить к своему логическому завершению. Официальное объявление Java, уже получившей широкое признание и подающей самые радужные надежды, должно было произойти на конференции SunWorld. Ожидалось, что

это будет короткое информационное объявление, так как главная цель этого мероприятия - UNIX-системы. Однако все произошло не так, как планировалось.

В 4 часа утра в день конференции, после длинных и сложных переговоров, Sun подписывает важнейшее соглашение. Вторая сторона - компания Netscape, основанная в апреле 1994 года Джеймсом Кларком (он уже сыграл роль в судьбе Oak два года, когда перехватил предложение от Time Warner) и Марком Андриссеном (создателем NCSA Mosaic). Эта компания являлась лидером рынка браузеров после того, как в декабре 1994 года вышла первая версия Netscape Navigator, которая была открыта для бесплатного некоммерческого использования, что позволило занять на тот момент 75% рынка.

23 мая 1995 года технология Java и HotJava были официально объявлены Sun [14], и тут же было сообщено, что новая версия самого п¿опулярного браузера Netscape Navigator 2.0 будет поддерживать новую технологию [15]. По сути, это означало, что отныне Java становится такой же неотъемлемой составляющей WWW, как и HTML. Во второй раз презентация закончилась бурными аплодисментами всех присутствующих. Победное шествие Java началось.

3. История развития Java

Теперь, когда за Java стояли не только несколько создателей, но еще и целая армия разработчиков, корпорация Sun имела возможность построить впечатляющие планы развития технологии.

3.1. Браузеры

Конечно, основная линия развития оставалась связанной с браузерами. Хотя Интернет только начинал наполняться все новыми технологиями, уже возникали проблемы совместимости. Под разными платформами были немного разные браузеры, различались даже шрифты. В результате автор мог создать красивую аккуратную страницу, которая расползалась у клиента.

С помощью Java веб-страницу можно наполнить не только обычным текстом, но и динамическими элементами - простыми видео-вставками типа вращающегося земного шара или Дьюка, машущего рукой (хотя сейчас такие задачи хорошо решает анимированный GIF, а в более сложных случаях - Macromedia Flash); интерактивные элементы типа вращающейся модели химической молекулы; бегущие строки, содержащие, например, биржевые индексы или прогноз погоды.

Но на самом деле Java - это больше, чем симпатичное украшение HTML. Поскольку это полноценный язык программирования, то с его помощью можно создавать сложный пользовательский интерфейс. В самой первой версии Java Development Kit (средство разработки на Java) был пример апплета, представляющий простейшие электронные таблицы. Вскоре появился текстовый редактор, позволяющий менять стиль и цвет текста. Конечно, были игровые апплеты, обучающие, моделирующие физические и иные системы. Например, клиент, сделавший заказ в магазине или отправивший посылку почтой, получал возможность следить за доставкой через Интернет.

В отличие от обычных программ апплеты получили "в наследство" важное свойство HTML страниц. Прочитав сегодня содержание страницы новостей, клиент не сохраняет ее на

своем компьютере, а на следующий день читает обновленное содержание. Точно также, скачав апплет и поработав с ним, можно его удалить, а в следующий раз получить более новую версию. Таким образом, программы появляются и исчезают с машины клиента безо всякого усилия, не требуются ни специальные знания, ни действия, при этом автоматически поддерживаются самые последние версии.

С другой стороны, пользователь уже не привязан к своему основному рабочему месту, в любом интернет-кафе можно открыть нужную веб-страницу и начать работу с привычными программами. И все это без каких-либо опасений "подцепить" вирус. Для разработчиков было очень привлекательно, что их программы через день после выпуска могут увидеть самые разные пользователи по всему миру, независимо от того, какой компьютер, операционную систему и браузер они используют. Хотя браузер на стороне клиента должен поддерживать Java, как уже говорилось, пользователям предлагался HotJava, доступный на любой платформе. Самый популярный в то время Netscape Navigator, начиная с версии 2.0, также содержал Java. Однако, сегодня, как известно самый распространенный браузер - Microsoft Internet Explorer.

Компания Microsoft, добившись ошеломляющего успеха в области программного обеспечения для персональных компьютеров, стала (и в целом остается до сих пор) основным конкурентом в этой области для Sun, IBM, Netscape и других. Если в начале девяностых основные усилия Microsoft были направлены на операционную систему Windows и офисные приложения (MS Office), то в середине десятилетия стало очевидно, что пора всерьез заняться Internet. В начале 1995 года Билл Гейтс опубликовал планы объявления "войны" Netscape с целью занять такое же монопольное положение в WWW, как и в области операционных систем для персональных компьютеров. И когда вскоре Netscape подписывает лицензионное соглашение с Sun, Microsoft оказалась в трудной позиции.

Internet Explorer 2.0 был настолько непривлекательным, что никто не верил, что он может составить какую-нибудь заметную конкуренцию Netscape Navigator. А это значит, что новая версия IE 3.0 должна уметь все, что умеет только что вышедший NN 2.0. Поэтому 7 декабря 1995 года Microsoft объявляет о своем желании лицензировать Java, а в марте 1996 года соглашение о лицензировании подписано. Самая крупная компания по производству программного обеспечения была вынуждена поддерживать своего, возможно, самого опасного конкурента.

Сейчас мы имеем возможность оглянуться назад и оценить последствия прошлых событий. Теперь уже очевидно, что Microsoft полностью удалось осуществить свой план. Если Netscape Navigator 3.x еще соблюдал лидирующее положение, то Netscape 4.x уже начал уступать Internet Explorer 4.x. NN 5.x так и не вышел, а NN 6.x стал очередным разочарованием для бывших поклонников "Навигатора". Сейчас вышла версия 7.0, однако она не занимает серьезной доли рынка, в то время как Internet Explorer 5.0, 5.5 и 6.0 используют более 95% пользователей.

Забавно, что многие ожесточенно обвиняли Microsoft в том, что она боролась с Netscape нерыночными средствами. Однако сравним действия конкурентов. Среди многих шагов, предпринятых Microsoft для победы, была и поддержка независимой организации W3C, которая руководила разработкой нового стандарта HTML 3. Вначале Netscape считался локомотивом индустрии, постоянно развивая и модернизируя HTML, который изначально вообще-то не предназначался для графического оформления текста. Но когда за дело взялась Microsoft, она, вложив большое количество денег и людских ресурсов, смогла утвердить стандарты, которые отличались от уже реализованных в Netscape Navigator,

причем отличия порой были чисто формальными. В результате оказалось, что страницы, сделанные в соответствии с W3C спецификациями, отображались в Navigator искаженно. Немаловажно и то, что NN необходимо было скачивать (пусть и бесплатно) и устанавливать вручную, а IE быстро стал встроенным компонентом Windows, сразу готовым к использованию (и от которого, к слову, избавиться нельзя было принципиально).

А каким образом Netscape смог добиться лидирующего положения? В свое время подобными же методами компания пытался (успешно, в конце концов) выдавить с рынка NCSA Mosaic. Тогда HTML был особенно беден интересными возможностями, а потому интересные инновации, поддерживаемые Navigator'ом, сразу привлекали внимание разработчиков и пользователей. Однако такие страницы совершенно неправильно отображались в Mosaic, что склоняло его пользователей к переходу на решения компании Netscape.

В результате в связи с забвением Netscape и его Navigator многие вздохнули с облегчением. Хотя, безусловно, потеря конкуренции на рынке и воцарение такого опасного монополиста как Microsoft, никогда не идет на пользу конечным пользователям, однако, многие устали от "войны стандартов", когда и так небогатые возможности HTML приходилось изощренно подгонять таким образом, чтобы страницы выглядели одинаково в обоих браузерах.

Про HotJava, к сожалению, особенно сказать нечего. Некоторое время Sun поддерживала этот продукт и добавила возможность визуально создавать веб-страницы без знания HTML. Однако создать конкурентоспособный браузер не удалось, и вскоре развитие HotJava было остановлено. Сейчас еще можно скачать и посмотреть последнюю версию 3.0.

И последнее, на чем стоит остановиться, - это язык Java Script, который также весьма распространен и который до сих пор многие связывают с Java, видимо, по причине схожести имен. Впрочем, некоторые общие особенности у них действительно есть.

4 декабря 1995 года компании Netscape и Sun совместно объявляют новый "язык сценариев" (scripting language) Java Script. Как следует из пресс-релиза это открытый, кросс-платформенный объектный язык сценариев для корпоративных сетей и Интернета. Код Java Script описывается прямо в HTML тексте (хотя возможно и подгружать его из отдельных файлов с расширением .js). Этот язык предназначен для создания приложений, которые связывают объекты и ресурсы на клиентской машине или на сервере. Таким образом, Java Script с одной стороны расширяет и дополняет HTML, а с другой стороны - дополняет Java. С помощью Java пишутся объекты-апплеты, которыми можно управлять через язык сценариев.

Общие свойства Java Script и Java:

· легкость в освоении. По этому параметру Java Script сравнивают с Visual Basic - чтобы использоваться эти языки, серьезный опыт программирования не требуется.

· кросс-платформенность. Код Java Script выполняется браузером. Подразумевается, что браузеры на разных платформах должны обеспечивать одинаковую функциональность для страниц, использующих язык сценариев. Однако, это выполняется примерно в той же степени, что и поддержка самого HTML - различий все же очень много.

· открытость. Спецификация языка открыта для использования и обсуждения сообществом разработчиков.

· все перечисленные свойства позволяют утверждать, что Java Script хорошо приспособлен для интернет-программирования.

· синтаксисы языков Java Script и Java очень похожи. Впрочем, они также довольно сильно напоминают язык C.

· язык Java Script не объектно-ориентированный (хотя некоторые аспекты OO подхода поддерживаются), но позволяет использование различных объектов, предоставляемых браузером.

· похожая история появления и развития. Оба языка были объявлены компаниями Sun и Netscape с интервалом в несколько месяцев. Вышедший вскоре после это Netscape Navigator 2.0 поддерживал обе новые технологии. Есть предположение, что само название Java Script было дано для того, чтобы воспользоваться большой популярностью Java, либо для того, чтобы еще больше расширить понятие "платформа Java". Вполне вероятно, что основную работу по разработке языка провела именно Netscape.

Несмотря на большое количество схожих характеристик, Java и Java Script - совершенно различные языки, и в первую очередь - по назначению. Если изначально Java позиционировалась как язык для создания интернет-приложений (апплетов), то сейчас уже совершенно ясно, что Java - это полноценный язык программирования. Что касается Java Script, то он полностью оправдывает свое название языка сценариев, оставаясь расширением HTML. Впрочем, расширением довольно мощным, так как любители этой технологии ухитряются создавать вполне серьезные приложения, такие как 3D игры от первого лица (в сильно упрощенном режиме, естественно), хотя это скорее случай из области курьезов.

В заключение отметим, что код Java Script, исполняющийся на клиенте, оказывается доступен всем в открытом виде, что затрудняет охрану авторских прав. С другой стороны, из-за отсутствия полноценной поддержки объявления новых типов программы со сложной функциональностью зачастую оказываются слишком запутанными для того, чтобы ими могли воспользоваться другие.

3.2. Сетевые компьютеры

Когда стало понятно, что новая технология пользуется небывалым спросом, естественным желанием было укрепить и развить успех и распространенность Java. Для того чтобы Java не разделила судьбу NeWS (эта оконная система упоминалась в начале главы, она не получила развития, проиграв конкуренцию X Window), компания Sun старалась наладить сотрудничество с третьими фирмами для производства различных библиотек, средств разработчика, инструментариев. 9 января 1996 года было сформировано новое подразделение JavaSoft, которое и занялось разработкой новых Java-технологий и продвижением их на рынок. Главная цель - появление все большего количества самых разных приложений, написанных на этой платформе. Например, 1 июля 1997 года было объявлено, что ученые NASA (National Aeronautics and Space Administration, государственная организация США, занимающаяся исследованием космоса) с помощью Java-апплетов управляют роботом, изучающим поверхность Марса ("Java помогает делать историю!").

Пора остановиться подробнее на том, почему по отношению к Java используется этот термин - "платформа", чем Java отличается от обычного языка программирования?

Как правило, платформой называют сочетание, во-первых, аппаратной архитектуры ("железо"), которая определяется типом используемого процессора (Intel x86, Sun SPARC, PowerPC и др.), и, во-вторых, операционной системой (MS Windows, Sun Solaris, Linux, Mac OS и др.). При написании программ разработчик всегда пользуется средствами целевой

платформы для доступа к сети, поддержки потоков исполнения, работы с графическим пользовательским интерфейсом (GUI) и другим возможностям. Конечно, различные платформы в силу технических, исторических и других причин поддерживают различные интерфейсы (API, Application Programming Interface), а значит и программа может исполняться только под той платформой, под которую она была написана.

Однако часто заказчикам требуется одна и та же функциональность, а платформы они используют разные. Задача портирования приложений стоит перед разработчиками давно. Редко удается перенести сложную программу без существенной переделки, очень часто различные платформы слишком по-разному поддерживают многие возможности (например, операционная система Mac OS традиционно использует однокнопочную мышь, в то время как Windows изначально рассчитывалась на двухкнопочную).

А значит и языки программирования должны быть изначально ориентированы на какую-то конкретную платформу. Синтаксис и основные концепции легко распространить на любую систему (хотя это и не всегда эффективно), но библиотеки, компилятор и, естественно, бинарный, исполняемый код специфичен для каждой платформы. Так было с самого начала компьютерных вычислений, а потому лишь немногие, действительно удачные программы поддерживались сразу на нескольких системах, что приводило к некоторой изоляции миров программного обеспечения для различных операционных систем.

Было бы странно, если с развитием компьютерной индустрии разработчики не попытались создать универсальную платформу, под которой могли работать все программы. Особенно такому шагу способствовало бурное развитие Глобальной Сети Интернет, которая объединила пользователей независимо от типа используемых процессоров и операционных систем. Именно поэтому создатели Java задумали разработать не просто еще один язык программирования, а универсальную платформу для исполнения приложений, тем более что изначально Oak планировался для различных бытовых приборов, от которых ждать совместимости не приходится.

Каким же образом можно "сгладить" различия и многообразие операционных систем? Способ не новый, но эффективный - виртуальная машина. Приложения на языке Java исполняются в специальной, универсальной среде, которая называется Java Virtual Machine. JVM - это программа, которая пишется специально для каждой реальной платформы, чтобы с одной стороны скрыть все ее особенности, а с другой - предоставить единую среду исполнения для Java-приложений. Фирма Sun и ее партнеры создали JVM практически для всех современных операционных систем. Когда говорится о браузере с поддержкой Java, также подразумевается, что в нем имеется встроенная виртуальная машина.

Подробнее JVM рассматривается ниже, но необходимо сказать, что компания Sun прикладывала усилия, чтобы сделать эту машину вполне реальной, а не только виртуальной. 29 мая 1996 года объявляется операционная система Java OS (финальная версия выпущена в марте следующего года). Согласно пресс-релизу - "возможно, самая небольшая и быстрая операционная система, поддерживающая Java". Действительно, единственной целью ее создателей была возможность исполнять Java-приложения на широком спектре устройств - сетевые компьютеры, карманные компьютеры (PDA), принтеры, игровые приставки, мобильные телефоны и многие другие. Ожидалось, что Java OS будет реализована на всех аппаратных платформах. Это было необходимо для изначальной цели создателей Java - легкость добавления новой функциональности и совместимости в любые электрические приборы, которыми пользуется современный потребитель.

Это был первый шаг, распространяющий платформу Java на один уровень вниз - на уровень операционных систем. Предполагалась сделать и следующий шаг - создать аппаратную архитектуру, центральный процессор, который бы напрямую выполнял инструкции Java безо всякой виртуальной машины. Устройство с такой реализацией стало бы полноценным Java-устройством "на 100%".

Кроме бытовых приборов, компания Sun позиционировала такое решение и для компьютерной индустрии - сетевые компьютеры должны были заменить разнородные платформы персональных рабочих станций. Такой подход хорошо ложился в центральную концепцию Sun, выраженную в лозунге "Сеть - это компьютер". Возможности одного компьютера никогда не сравнятся с возможностями сети, объединяющей все ресурсы компании, а тем более - всего мира. Наверное, сегодня это уже очевидно, но во времена, когда WWW еще не опутала весь мир, идея была революционной.

Если же пытаться построить многофункциональную сеть, то к ее рабочим станциям предъявляются совсем другие требования - им не нужно быть особенно мощными - вычислительные задачи можно переложить на сервера. Более того, это особенно выгодно, так как позволит централизовать поддержку и обновление программного обеспечения, а также позволит сотрудникам не быть привязанным к своим рабочим местам. Достаточно войти с любого терминала в сеть, авторизоваться - и можно продолжать работу с того места, на котором она была оставлена. Это можно сделать в кабинете, зале для презентаций, кафе, в кресле самолета, дома - где угодно!

Кроме своих несомненных удобств, это начинание было с большим энтузиазмом поддержано индустрией и в силу того, что оно являлось сильнейшим оружием в борьбе с крупнейшей корпорацией-производителем программного обеспечения Microsoft. Тогда (да и сейчас) самой распространенной платформой являлась операционная система Windows на базе процессоров Intel (с чьей-то легкой руки теперь многими называемая Wintel). Этим компаниям удалось создать замкнутый круг, гарантирующий успех - все пользовались их платформой, так как под нее написано больше всего программ, что в свою очередь склоняло разработчиков создавать новые продукты именно для платформы Wintel. Поскольку корпорация Microsoft всегда очень агрессивно развивала свое преимущество в области персональных компьютеров (вспомним, как Netscape Navigator безнадежно проиграл конкуренцию MS Internet Explorer), это не могло не вызывать сильное беспокойство других представителей компьютерной индустрии. Понятно, что концепция сетевых компьютеров легко устраняла бы преимущества Wintel в случае широкого распространения. Разработчики и пользователи просто перестали бы задумываться, что находится внутри их рабочей станции, также как домашние потребители не имеют представления, на каких микросхемах собран их мобильный телефон или видеомагнитофон.

Выше рассказывалось, как и почему Microsoft лицензировала Java, хотя, казалось бы, этот шаг лишь способствовал опасному распространению новой технологии, ведь Internet Explorer завоевывал все большую популярность. Однако вскоре разразился судебный скандал. 30 сентября 1997 года вышел новый IE 4.0, а уже 7 октября Sun объявляет, что этот продукт не проходит тесты на соответствие со спецификацией виртуальной машины. 18 ноября Sun обращается в суд, чтобы запретить использование логотипа "Совместимый с Java" ("Java compatible") для MS IE 4.0. Оказалось, что Microsoft слегка "улучшила" язык Java, добавив несколько новых ключевых слов и библиотек. Не то чтобы это были сверхмощные расширения, однако достаточно привлекательные, чтобы заметная часть разработчиков начала ее использовать. К счастью, в Sun быстро осознали всю степень опасности такого шага. Java могла перестать быть универсальной платформой, для которой

верен знаменитый девиз "Write once, run everywhere" ("Написано однажды, работает везде"). В таком случае она потеряла бы основу своего успеха, превратившись всего лишь в "еще один язык программирования".

Компания Sun смогла отстоять свою технологию. 24 марта 1998 года суд согласился с требованиями компании (конечно, это было только предварительное решение, дело завершилось лишь 23 января 2001 года, Sun получил компенсацию в 20 миллионов долларов и добился выполнения лицензионного соглашения), а уже 12 мая Sun снова выступает с требованием обязать Microsoft включить полноценную версию Java в Windows 98 и другие программные продукты. Эта история продолжается до сих пор с переменным успехом сторон. Например, Microsoft исключила из виртуальной машины Internet Explorer'a библиотеку java.rmi, позволяющей легко создавать распределенные приложения, пытаясь привлечь разработчиков к DCOM - технологии, жестко привязанной к платформе Win32. В ответ многие компании стали распространять специальное дополнение (patch), устраняющее этот недостаток. В результате Microsoft остановила свою поддержку Java на версии 1.1, которая на данный момент является устаревшей и не имеет многих полезных возможностей. Это в свою очередь практически остановило широкое распространение апплетов, кроме случаев либо совсем несложной функциональности (типа бегущей строки или диалога с несколькими полями ввода и кнопками), либо приложений для внутренних сетей корпораций. Для последнего случая Sun выпустил специальный продукт Java Plug-in, который встраивается в MS IE и NN, позволяя им исполнять апплеты на основе Java самых последних версий, причем полное соответствие спецификациям гарантируется (первоначально продукт назывался Java Activator и впервые был объявлен 10 декабря 1997 года). На данный момент Microsoft то включает, то исключает Java из своей операционной системы Windows XP, видимо, пытаясь найти самый выгодный для себя вариант.

Что же касается сетевых компьютеров и Java OS, увы, они не смогли найти своих потребителей. Видимо, обычные персональные рабочие станции в совокупности с JVM требуют гораздо меньше технологических и маркетинговых усилий, и при этом вполне успешно справляются с прикладными задачами. А Java, в свою очередь, стала позиционироваться для создания сложных серверных приложений.

3.3. Платформа Java

Итак, Java обладает длинной и непростой историей развития, однако настало время рассмотреть, что же получилось у создателей, какими свойствами отличается эта технология.

Самое широко известное, и в тоже время вызывающее самые бурные споры, свойство много- или кросс-платформенности. Уже описывалось, что оно достигается за счет использования виртуальной машины JVM, которая является обычной программой, исполняемой операционной системой и предоставляющей все необходимые возможности Java-приложениям. Поскольку все параметры JVM специфицированы, то остается единственная задача - реализовать виртуальные машины на всех существующих и используемых платформах.

Наличие виртуальной машины определяет многие свойства Java, однако сейчас остановимся на следующем вопросе - является Java языком компилируемым или интерпретируемым? На самом деле, используются оба подхода.

Исходный код любой программы на языке Java представляется обычными текстовыми файлами, которые могут быть созданы в любом текстовом редакторе или специализированном средстве разработки и имеют расширение .java. Эти файлы подаются на вход Java-компилятора, который транслирует их в специальный Java байт-код. Именно этот компактный и эффективный набор инструкций поддерживается JVM и является неотъемлемой частью платформы Java.

Результат работы компилятора сохраняется в бинарных файлах с расширением .class. Java-приложение, состоящее из таких файлов, подается на вход виртуальной машине, которая начинает их исполнять, или интерпретировать, так как сама является программой.

Многие разработчики на раннем этапе жестко критиковали смелый лозунг Sun "Write once, run everywhere", обнаруживая все больше и больше несоответствий и нестыковок на различных платформах. Однако надо признать, что они были просто слишком нетерпеливыми. Java только появилась на свет, а первые версии спецификаций были недостаточно исчерпывающими. Кроме того, разработчики виртуальных машин также являются обычными людьми, и порой делают ошибки в своих продуктах.

Очень скоро Sun пришел к выводу, что просто свободно публиковать свои спецификации (что уже делалось задолго до Java) недостаточно. Необходимо еще и создавать специальные процедуры проверки новых продуктов на соответствие стандартам. Первый такой тест для JVM содержал всего около 600 проверок, через год их число выросло до десяти тысяч, и с тех пор все время увеличивается (именно его в свое время не смог пройти MS IE 4.0). Также, безусловно, авторы виртуальных машин все время совершенствовали их, устраняя ошибки и оптимизируя работу. Все-таки любая, даже очень хорошо задуманная, технология требует времени для создания высококачественной реализации. Аналогичный путь развития сейчас проходит Java 2 Micro Edition (J2ME), но об этом позже.

Следующим по важности является объектная ориентированность Java, что всегда упоминается во всех статьях и пресс-релизах. Важность самого ООП обсуждается в следующей главе, однако важно подчеркнуть, что в Java практически все реализовано в виде объектов - потоки выполнения (threads) и потоки данных (steams), работа с сетью, работа с изображениями, с пользовательским интерфейсом, обработка с ошибками и т.д. В конце концов, любое приложение на Java - это набор классов, описывающих новые типы объектов.

Подробное рассмотрение объектной модели Java проводится на протяжении всего курса, однако обозначим наиболее значимые особенности. Во-первых, создатели отказались от множественного наследования. Было решено, что оно слишком усложняет и запутывает программы. В языке используется альтернативный подход - специальный тип "интерфейс". Он подробно рассматривается в соответствующей главе.

Далее, в Java применяется строгая типизация. Это означает, что любая переменная и любое выражение имеет тип, известный уже на момент компиляции. Такой подход применен для упрощения выявления проблем, ведь компилятор сразу сообщает об ошибках и указывает их расположение в коде. Поиск же исключительных ситуаций (exceptions - так в Java называются некорректные ситуации) во время исполнения программы (runtime) потребует сложного тестирования, причем причина, породившая дефект, может обнаружиться в совсем другом классе. Таким образом, нужно прикладывать дополнительные усилия при написании кода, зато существенно повышается его надежность (а это одна из основополагающих целей, для которых и создавался новый язык).

В Java есть всего 8 типов данных, которые не являются объектами. Они были определены с самой первой версии и никогда не менялись. Это 5 целочисленных типов: byte, short, int, long, а также к ним относят символьный char. Затем 2 дробных типа float и double, и наконец булевский тип boolean. Такие типы называются простыми или примитивными (от английского primitive), и они подробно рассматриваются в главе, посвященной типам данных. Все остальные - объектные или ссылочные (англ. reference).

Синтаксис Java почему-то многих ввел в заблуждение. Он действительно создан на основе синтаксиса языков C/C++, так что если посмотреть на исходный код программ, написанных на этих языках и на Java, то не сразу удается понять, какая из них на каком языке написана. Это почему-то дало основание делать утверждения типа "Java - это упрощенный C++ с дополнительными возможностями, такими как garbage collector". Автоматический сборщик мусора (garbage collector) обсуждается чуть ниже, но считать что Java такой же язык, как и C++, - большое заблуждение.

Конечно, разрабатывая новую технологию, авторы Java опирались на широко распространенный язык программирования в силу целого ряда причин. Во-первых, они сами на тот момент считали C++ своим основным инструментом. Во-вторых, зачем придумывать что-то новое, когда есть хорошо подходящее старое? Наконец, очевидно, что незнакомый синтаксис отпугнет разработчиков и существенно осложнит внедрение нового языка, а ведь Java должна была максимально быстро получить широкое распространение. Поэтому синтаксис был лишь слегка упрощен, чтобы избежать слишком запутанных конструкций.

Но, как уже описывалось, C++ принципиально не годился для новых задач, которые поставили себе разработчики из компании Sun, поэтому идеология, модель Java была построена заново, причем в соответствии с совсем другими целями. Дальнейшие главы будут постепенно раскрывать конкретные различия.

Что же касается объектной модели, то она скорее была построена по образцу таких языков, как Smalltalk от IBM или разработанный еще в 60-е годы в Норвежском Вычислительном Центре язык Simula, на который ссылается сам создатель Java Джеймс Гослинг.

Другое немаловажное свойство Java - легкость в освоении и разработке - также получило неоднозначное трактование. Действительно, авторы позаботились избавить программистов от наиболее распространенных ошибок, которые порой допускают даже опытные разработчики на C/C++. И первое место здесь занимает работа с памятью.

В Java с самого начала был введен механизм автоматической сборки мусора (от английского garbage collector). Предположим, программа создает некоторый объект, работает с ним, а дальше наступает момент, когда он больше уже не нужен. Необходимо освободить занимаемую память, чтобы не мешать операционной системе нормально функционировать. В C/C++ это необходимо делать явным образом из программы. Понятно, что при таком подходе есть две опасности - либо удалить объект, который еще кому-то необходим (и если к нему действительно произойдет обращение, то возникнет ошибка), либо не удалить объект, ставший ненужным, а это означает утечку памяти, то есть программа начинает потреблять все большое количество оперативной памяти.

При разработке на Java программист вообще не думает об освобождении памяти. Виртуальная машина сама подсчитывает количество ссылок на каждый объект, и если оно становится равным нулю, то такой объект помечается для обработки garbage collector. Таким образом, программист должен следить лишь за тем, чтобы не оставалось ссылок на ненужные объекты. Сборщик мусора - это фоновый поток исполнения, который регулярно

просматривает существующие объекты, и удаляет уже не нужные. Из программы никак нельзя повлиять на работу garbage collector, можно только явно инициировать его очередной проход с помощью стандартной функции. Ясно, что это существенно упрощает разработку программ, особенно для начинающих программистов.

Однако опытные разработчики были недовольны тем, что они не могут полностью контролировать все, что происходит с их системой. Нет точной информации, когда именно будет удален объект, ставший ненужным, когда начнет работать (а значит и занимать системные ресурсы) поток сборщика мусора и т.д. Но, при всем уважении к опыту таких программистов, необходимо отметить, что подавляющее количество сбоев программ, написанных на C/C++, приходится именно на некорректную работу с памятью, причем порой это случается даже с широко распространенными продуктами весьма серьезных компаний.

Кроме того, особый упор делался на легкое освоение новой технологии. Как уже говорилось, ожидалось (и эти ожидания оправдались, что подтверждает правильность выбранного пути!), что Java должна получить максимально широкое применение, даже в таких компаниях, где никогда до этого не занимались программированием на таком уровне (бытовая техника типа тостеров и кофеварок, создание игр и других приложений для сотовых телефонов и т.д.). Был и целый ряд других соображений. Продукты для обычных пользователей, а не профессиональных программистов, должны быть особенно надежны. Интернет стал Всемирной Сетью за счет прихода непрофессиональных пользователей, а возможность создавать апплеты для них не менее привлекательна. Им требовался простой инструмент для создания надежных приложений.

Наконец, Интернет-бум 90-х годов набирал обороты и выдвигал новые, более жесткие требования на сроки разработки. Многолетние проекты, которые были обычным делом в прошлом, перестали отвечать потребностям заказчиков, новые системы надо было создавать максимум за год, а то и считанные месяцы.

Кроме введения garbage collector, были предприняты и другие шаги для облегчения разработки. Некоторые уже упоминались - отказ от множественного наследования, упрощение синтаксиса и др. Возможность создание многопоточных приложений было внесено с самой первой версии Java (исследования показали, что это очень удобно для пользователей, а существующие стандарты опираются на телетайпные системы, которые устарели много лет назад). Другие особенности будут рассмотрены в следующих главах. Однако то, что создание и поддержка систем действительно проще на Java, чем на C/C++, давно является общепризнанным фактом. Впрочем, все-таки эти языки созданы для разных целей, и каждый имеет свои неоспоримые преимущества.

Следующее важное свойство Java - безопасность. Изначальная нацеленность на распределенные приложение, и в особенности решение исполнять апплеты на клиентской машине, сделали вопрос защиты одним из самых приоритетных. При работе любой виртуальной машины Java действует целый комплекс мер. Далее приводится лишь краткое описание некоторых из них.

Во-первых, это правила работы с памятью. Уже говорилось, что очищение памяти производится автоматически. Резервирование ее также определяется JVM, а не компилятором или явным образом из программы, разработчик может лишь указать, что он хочет создать еще один новый объект. Указатели по физическим адресам отсутствуют принципиально.

Во-вторых, наличие виртуальной машины-интерпретатора значительно облегчает отсечение опасного кода на каждом этапе работы. Сначала байт-код загружается в систему, как правило, в виде class-файлов. JVM внимательно анализирует, что все они подчиняются общим правилам безопасности Java, а не были созданы злоумышленниками с помощью каких-то других средств (или не были искажены при передаче). Затем во время исполнения программы, интерпретатор легко может проверить каждое действие на допустимость. Возможности классов, которые были загружены с локального диска или по сети, существенно различаются (конечный пользователь легко может назначать или отменять конкретные права). Например, апплет по умолчанию никогда не получит доступ к локальной файловой системе. Такие встроенные ограничения есть и во всех стандартных библиотеках Java.

Наконец, существует механизм подписания апплетов и других приложений, загружаемых по сети. Специальный сертификат гарантирует, что пользователь получил код именно в том виде, в каком его выпустил производитель. Это, конечно, не дает дополнительных средств защит, но позволяет клиенту либо отказаться от работы с приложениями ненадежных производителей, либо сразу увидеть, что в программу внесены неавторизованные изменения. В худшем случае он знает, кто ответственен за причиненный ущерб.

Совокупность описанных свойств Java позволяет утверждать, что язык весьма приспособлен для разработки интернет- и интранет-(внутренние сети корпораций) приложений.

Наконец, важная отличительная особенность Java - это ее динамичность. Язык очень удачно задуман, в его развитии участвуют сотни тысяч разработчиков и многие крупные компании. Основные этапы этого развития кратко освещены в следующем разделе.

Итак, подведем итоги. Java-платформа обладает следующими преимуществами:

· переносимость, или кросс-платформенность;

· объектная ориентированность, создана эффективная объектная модель;

· привычный синтаксис C/C++;

· встроенная и прозрачная модель безопасности;

· ориентация на интернет-задачи, сетевые распределенные приложения;

· динамичность, легкость развития и добавления новых возможностей;

· легкость в освоении.

Но не следует считать, что более легкое освоение означает, что изучать язык не нужно вовсе. Чтобы писать действительно хорошие программы, создавать большие сложные системы, необходимо четкое понимание всех базовых концепций Java и используемых библиотек. Именно этому и посвящен данный курс.

Сразу оговоримся, что под продуктами здесь понимаются программные решения от компании Sun, являющиеся "образцами реализации" (reference implementation).

Итак, впервые Java была объявлена 23 мая 1995 года. Основными продуктами, доступными на тот момент в виде бета-версий, были:

· Java language specification, JLS, спецификация языка Java (описывающая лексику, типы данных, основные конструкции, и т.д.);

· спецификация JVM;

· Java Development Kit, JDK - средство разработчика, состоящее в основном из утилит, стандартных библиотек классов и демонстрационных примеров.

Спецификация языка была составлена настолько удачно, что практически без изменений используется по сей день. Конечно, было внесено большое количество уточнений, более подробных описаний, были добавлены и некоторые новые возможности (например, объявление внутренних классов), однако основные концепции остаются неизменными. Данный курс в большой степени опирается именно на спецификацию языка.

Спецификация JVM предназначена в первую очередь для создателей виртуальных машин, а потому практически не используется Java-программистами.

JDK долгое время было базовым средством разработки приложений. Оно не содержит никаких текстовых редакторов, а оперирует только с уже существующими java-файлами. Компилятор представлен утилитой javac (java compiler) . Виртуальная машина реализована программой java. Для тестовых запусков апплетов есть специальная утилита appletviewer. Наконец, для автоматической генерации документации на основе исходного кода прилагается средство javadoc.

Первая версия содержала всего 8 стандартных библиотек:

· java.lang - базовая классы, необходимые для работы любого приложения (название - сокращение от language);

· java.util - многие полезные вспомогательные классы;

· java.applet - классы для создания апплетов;

· java.awt, java.awt.peer - библиотека для создания графического интерфейса пользователя (GUI), называется Abstract Window Toolkit, AWT. Подробно описана в соответствующей главе.

· java.awt.image - дополнительные классы для работы с изображениями;

· java.io - работа с потоками данных (streams) и с файлами;

· java.net - работа с сетью.

Как видно, все библиотеки начинаются с java, именно они являются стандартными. Все остальные (начинающиеся с com, org и др.) могут меняться в любой версии без поддержки совместимости.

Финальная версия JDK 1.0 была выпущена в январе 1996 года.

Сразу поясним систему именования версий. Обозначение версии состоит из трех цифр. Первой пока всегда стоит 1. Это означает, что поддерживается полная совместимость между всеми версиями 1.x.x. То есть, программа, написанная на более старом JDK, всегда успешно выполнится на более новом. По возможности соблюдается и обратная совместимость - если программа откомпилирована более новым JDK, а никакие новые библиотеки не использовались, то в большинстве случаев старые виртуальные машины смогут выполнить такой код.

Вторая цифра изменилась от 0 до 4 (последняя на данный момент). В каждой версии происходило существенное расширение стандартных библиотек (212, 504, 1781, 2130 и

2738 - количество классов и интерфейсов с 1.0 по 1.4), а также добавлялись некоторые новые возможности в сам язык. Менялись и утилиты, входящие в JDK.

Наконец, третья цифра означает развитие одной версии. В языке или библиотеках ничего не меняется, лишь устраняются ошибки, производится оптимизация, могут меняться (добавляться) аргументы утилит. Так, последняя версия JDK 1.0 - 1.0.2.

Хотя с развитием версии 1.x ничего не удаляется, конечно, какие-то функции или классы устаревают. Они объявляются deprecated, и хотя они будут поддерживаться до объявления 2.0 (а про нее пока ничего не было слышно), пользоваться ими не рекомендуется.

Вместе с первым успехом JDK 1.0 пришла и критика. Основные недостатки, обнаруженные разработчиками, были следующими. Во-первых, конечно, производительность. Первая виртуальная машина работала очень медленно. Это связано с тем, что JVM по сути интерпретатор, который работает всегда медленнее, чем исполняется откомпилированный код. Однако, успешная оптимизация, устранившая этот недостаток, была еще впереди. Также отмечались довольно бедные возможности AWT, отсутствие работы с базами данных и другие.

В декабре 1996 года объявляется новая версия JDK 1.1, сразу выкладывается для свободного доступа бета-версия. В феврале 1997 года выходит финальная версия. Что было добавлено в новом выпуске Java?

Конечно, особое внимание было уделено производительности. Во-первых, многие части виртуальной машины были оптимизированы и переписаны с использованием Assembler, а не C, как до этого. Кроме этого, с октября 1996 года Sun развивает новый продукт - Just-In-Time компилятор, JIT. Его задача - транслировать Java байт-код программы в "родной" код операционной системы. Таким образом, время запуска программы увеличивается, но зато выполнение может ускоряться в некоторых случаях до 50 раз! С июля 1997 года появляется реализация под Windows, и JIT стандартно входит в JDK с возможностью отключения.

Были добавлены многие новые важные возможности. JavaBeans - технология, объявленная еще в 1996 году, позволяет создавать визуальные компоненты, которые легко интегрируются в визуальные средства разработки. JDBC (Java DataBase Connectivity) обеспечивает доступ к базам данных. RMI (Remote Method Invocation) позволяет легко создавать распределенные приложения. Были улучшены поддержка национальных языков и безопасность.

За первые 3 недели JDK 1.1 был скачан более 220.000 раз, менее чем через год - более 2-х миллионов раз. На данный момент версия 1.1 считается полностью устаревшей, и ее развитие остановилось на 1.1.8. Однако из-за того, что самый распространенный браузер MS IE до сих пор поддерживает только эту версию, она продолжает использоваться для написания небольших апплетов.

Кроме этого, с 11 марта 1997 года компания Sun начала предлагать Java Runtime Environment, JRE (среду выполнения Java). По сути дела это минимальная реализация виртуальной машины, необходимая для исполнения Java-приложений, без компилятора и других средств разработки. Если пользователь хочет только запускать программы, это именно то, что ему нужно.

Как видно, самым главным недостатком осталась слабая поддержка графического интерфейса пользователя (GUI).В декабре 1996 года компании Sun и Netscape объявляют новую библиотеку IFC (Internet Foundation Classes), разработанную Netscape полностью на Java и предназначенную как раз для создания сложного оконного интерфейса. В апреле

1997 года объявляется, что компании планируют объединить технологии AWT от Sun и IFC от Netscape для создания нового продукта Java Foundation Classes, JFC, в который должны войти:

· улучшенный оконный интерфейс, который получил особое название Swing;

· реализация Drag-and-Drop.

· поддержка 2D графики, улучшенная работа с изображениями;

· Accessibility API для пользователей с ограниченными возможностями; и другие возможности. Компания IBM также поддержала разработку новой технологии. В июле 1997 года стала доступна первая версия JFC. Первоначально библиотеки назывались, например, com.sun.java.swing для компонент Swing. В марте 1998 года вышла финальная версия этой технологии. За полгода продукт был скачан более 500.000 раз.

Выход следующей версии Java 1.2 много раз откладывался, но в итоге она настолько превзошла предыдущую 1.1, что с этого момента ее и все последующие версии начали называть платформой Java 2 (хотя номера, конечно, продолжались отсчитываться как 1.x.x, см выше описание правил нумерации). Первая бета-версия стала доступной в декабре 1997 года, а финальная версия была выпущена 8 декабря 1998 года, и за первые восемь месяцев ее скачали более миллиона раз.

Список появившихся возможностей очень широк, поэтому перечислим наиболее значимые из них:

· серьезно переработанная модель безопасности, введены понятия политики (policy) и разрешения (permission);

· JFC стал стандартной частью JDK, причем библиотеки стали называться, например, javax.swing для Swing (название javax указывает, что до этого библиотека считалась расширением Java);

· полностью переработанная библиотека коллекций (collection framework) - классов для хранения набора объектов;

· Java Plug-in был включен в JDK;

· улучшения в производительности, глобализации (независимости от особенностей разных платформ и стран), защита от "проблемы-2000".

С февраля 1999 года исходный код самой JVM был открыт для бесплатного доступа всем желающим.

Самое же существенное изменение произошло спустя полгода после выхода JDK 1.2. 15 июня 1999 года на конференции разработчиков JavaOne компания Sun объявила о разделении развития платформы Java 2 на три направления:

· Java 2 Platform, Standard Edition (J2SE);

· Java 2 Platform, Enterprise Edition (J2EE);

· Java 2 Platform, Micro Edition (J2ME).

На самом деле, подобная классификация уже давно назрела, так различные спецификации и библиотеки насчитывались в количестве нескольких десятков штук, а потому нуждались в четкой структуризации. Кроме того, такое разделение облегчало развитие и вывод на рынок технологии Java.

J2SE предназначается для использования на рабочих станциях и персональных компьютерах. На самом деле, Standard Edition - основа технологии Java и прямое развитие JDK (средство разработчика было переименовано в j2sdk).

J2EE содержит все необходимое для создание сложных, высоконадежных, распределенных серверных приложений. Условно можно сказать, что Enterprise Edition - это набор мощных библиотек (например, Enterprise Java Beans, EJB) и пример реализации платформы (сервера приложений, Application Server), которая их поддерживает. Работа такой платформы всегда опирается на j2sdk.

J2ME является усечением Standard Edition для того, чтобы удовлетворять жестким аппаратным требованием небольших устройств, таких как карманные компьютеры и сотовые телефоны.

Далее развитие этих технологий происходит разными темпами. Если J2SE уже была доступен более полугода, то финальная версия J2EE вышла лишь в декабре 1999 года. Последняя версия j2sdk 1.2 на данный момент - 1.2.2.

Тем временем борьба за производительность продолжалась, и Sun пытался еще больше оптимизировать виртуальную машину. В марте 1999 года объявляется новый продукт - высокоскоростная платформа (engine) Java HotSpot. Была оптимизирована работа с потоками исполнения, существенно переработаны алгоритмы автоматического сборщика мусора (garbage collector) и многое другое. Ускорение действительно было очень существенным, всегда заметное даже невооруженным взглядом за несколько минут работы с Java-приложением.

Новая платформа может работать в двух режимах - клиентском и серверном. Режимы различались настройками и другими оптимизирующими алгоритмами. По умолчанию работа идет в клиентском режиме.

Развитие HotSpot продолжалось более года, пока в начале мая 2000 года высокопроизводительная JVM не вошла в состав новой версии J2SE. В эту версию было внесено еще множество улучшений и исправлений, но именно большой прогресс в ускорении работы стал ключевым изменением нового j2sdk 1.3 (последняя подверсия 1.3.1).

Наконец, последняя на данный момент версия J2SE 1.4 вышла в феврале 2002 года. Она была разработана для более полной поддержки веб-сервисов (web services). Поэтому основные изменения коснулись работы с XML (Extensible Markup Language). Другое революционное добавление - выражение assert, позволяющее в отладочном режиме проверять верность условий, что должно серьезно упростить разработку сложных приложений. Наконец, были добавлены классы для работы с регулярными выражениями.

За первые пять месяцев j2sdk 1.4 было скачано более двух миллионов раз. В августе 2002 года была уже предложена версия 1.4.1, остающаяся самой современной на данный момент.

В заключение для демонстрации уровня развития Standard Edition приведем стандартные диаграммы, описывающие все их составляющие технологии, из документации к версиям 1.3: и 1.4:

4. Заключение

В этой лекции Вы узнали, какая сложная ситуация сложилась в корпорации Sun в эпоху развития персональных компьютеров в конце 1990 года. Патрик Нотон в своем письме сумел выявить истинные причины такого положения и обозначить истинные цели для создания успешного продукта. Благодаря этому при поддержке Джеймса Гослинга начался проект Green. Одним из продуктов, созданных в рамках этого проекта, стала совершенно новая платформа Oak. Для ее продвижения Sun учредила дочернюю компанию FirstPerson, но настоящий успех пришел, когда платформу, переименовав в Java, сориентировали на применение в Интернете. Глобальная сеть появилась в апреле 1993 года с выходом первого браузера Mosaic 1.0 и завоевывала пользовательскую аудиторию с поразительной скоростью. Первым примером Java-приложений стали апплеты, запускаемые при помощи специально созданного браузера HotJava. Наконец, после почти 4-х летней истории создания и развития, Java была официально объявлена миру. Благодаря подписанию лицензионного соглашения с Netscape, это событие стало поистине триумфальным.

Были рассмотрены различные варианты применения Java и насколько удачно удалось их развить и воплотить в жизнь. Отдельно был описан язык Java Script, который, несмотря на сходство в названии, имеет не так много общих черт с Java. Подробно рассмотрены отличительные особенности Java. Описаны базовые продукты от Sun: JDK и JRE. Кратко освещена история развития версий платформы Java, включая добавляемые технологии и продукты.

5. Контрольные вопросы

1-1. Перечислите основные свойства и преимущества платформы Java. Что такое JVM?

а.) Основные свойства языка:

- Кросс-платформенный
- Объектно-ориентированный
- - Строгая типизация
  - Наличие 8 примитивных типов
  - Вместо множественного наследования введены интерфейсы
- Легкий в освоении и разработке
- - Привычный синтаксис Java, являющийся развитием синтаксиса C/C++
  - Автоматическая сборка мусора (garbage collection)
  - Повышенная надежность Java-программ
  - Изначальная поддержка многопоточной архитектуры
- Высоко защищенный
- - Отсутствие указателей

  - Обязательная проверка виртуальной машиной кода загружаемых классов и действий, выполняемых программой
  - Встроенные возможности (работа с SSL и др.)
  - Подпись апплетов
- Приспособленный к разработке интернет-приложений
- Динамичный и активно развивающийся

JVM – это Java Virtual Machine, виртуальная машина Java, интерпретирующая байт-код, описываемый в class-файлах. Ее применение необходимо для обеспечения кросс-платформенности, а также для безопасности, однако создает определенные проблемы в вопросе производительности.

1-2. Является ли язык Java компилируемым или интерпретируемым?

a.) Используются оба подхода. Исходный код сначала компилируется в байт-код, который затем интерпретируется виртуальной машиной.

1-3. Что такое механизм автоматической сборки мусора (garbage collector)?

a.) Этот механизм автоматически подсчитывает количество ссылок на каждый объект Java. Когда на объект больше не указывает ни одна ссылка, он удаляется из памяти, освобождая ресурсы для программы.

1-4. В чем сходства и различия Java и C/C++?

a.) Основным сходством является во многом похожий синтаксис. Различий гораздо больше – Java обладает свойством кросс-платформенности, применяет интерпретатор, использует отличную объектную модель и др.

1-5. Почему Java является платформой, а не языком программирования?

a.) Применяя язык программирования, всегда необходимо учитывать используемую платформу при обращении к аппаратным ресурсам, таким как файловая система, работа с сетью, потоки исполнения и другие.Java сама является полноценной платформой, предоставляя приложениям единый интерфейс и скрывая различия используемой операционной системы и аппаратной платформы.

1-6. Из-за каких опасений корпорация Sun вела многолетнюю судебную тяжбу с Microsoft после выхода MS Internet Explorer 4.0?

a.) Одним из важнейших свойств Java является кросс-платформенность, для чего необходимо абсолютное соблюдение спецификации языка. Если разные реализации платформы будут иметь различные свойства, то совместимость будет утеряна.

Фирма Microsoft в своем продукте Internet Explorer 4.0 реализовала виртуальную машину с нарушениями лицензионного соглашения, а именно как раз внесла не стандартизованные возможности, что ставило под угрозу ключевое преимущество Java.

1-7. Из чего состоит и в каком виде записывается программа, написанная на Java?

а.) Программа на языке Java состоит из объявления классов, записанных в текстовых файлах с расширением java. Затем с помощью компилятора генерируются бинарные .class-файлы, содержащие байт-код, который затем интерпретируется виртуальной машиной.Часто класс-файлы упаковываются в архивы (.jar или .zip).

1-8. Что можно сказать относительно скорости выполнения Java-программ, и какие шаги предпринимала компания Sun в этой направлении?

а.) Быстродействие программ первых версий Java оставляло желать много лучшего. Виртуальная машина является интерпретатором, и ее первые реализации страдали от отсутствия должной оптимизации. Однако даже в то время для Java это не было принципиальным недостатком, поскольку основные конкурентные преимущества заключались в другом.

Компания Sun, непрерывно оптимизируя JVM, предложила продукт JIT (Just-in-time), который перед исполнением программы транслировал ее в «родной» код применяемой платформы, что существенно ускоряло работу приложений. С версии 1.3 применяется виртуальная машина HotSpot, применяющая улучшенные алгоритмы сборки мусора и существенно улучшающая быстродействие.

1-9. Что такое апплет?

а.) Апплет – это приложение, написанное на Java, распространяемое, как правило, через сеть Интернет и выполняемое виртуальной машиной браузера. Являются частью HTML-страницы.

Обычно имеет небольшой размер и существенно ограничивается в правах доступа к системе клиента, чтобы обеспечить безопасность выполнения кода, полученного из открытых источников.

1-10. Когда было официально объявлено о Java?

а.) 23 мая 1995 года на конференции SunWorld.

1-11. Где в Интернете можно найти самую полную и актуальную информацию о Java-платформе?

а.) Официальный сайт Java – http://java.sun.com

1-12. Какова система версий в Java? Что означает название Java2? Какая последняя выпущенная версия Java?

а.) Номер версии состоит их трех чисел.

Первое число всех существующих на данный момент версий – единица. Такие версии всегда совместимы между собой, то есть, программы, написанные на основе более старых версий, всегда будут корректно исполняться более новыми версиями. По возможности сохраняется и обратная совместимость – программы, созданные на основе более новых версий, будут работать и для более старых версий, если они обладают всеми используемыми свойствами.

Второе число изменилось от 0 до 4 (версии 1.0 по 1.4). Каждая новая версия обладает новыми возможностями по сравнению с предыдущей. Java версии 1.2 настолько превосходила платформу 1.1, что, начиная с

нее, все последующие версии называли платформой Java2 (при этом сама система номеров версий не изменилась).

Третья цифра означает номер версии поддержки платформы. Не добавляются никакие новые возможности, но исправляются ошибки, возможна дополнительная оптимизация. Последняя версия на данный момент Java 1.4.0.

1-13. Что означает сообщение deprecated?

a.) На данный момент новые версии Java продолжают поддерживать все возможности старых, однако некоторые классы и методы становятся не рекомендованными к использованию и, возможно, будут удалены их последующих версий. В этом случае они называются deprecated, и в процессе компиляции будут выданы предупреждения о том, что таких конструкций необходимо по возможности избегать.

1-14. На какие три направления было поделено развитие Java вскоре после выхода Java2?

a.) В середине 1999 года было объявлено о разделении Java на три направления:

- J2SE – Java2 Standard Edition, основа технологии Java, прямое развитие JDK
- J2EE – Java2 Enterprise Edition, платформа для создания сложных, распределенных, высоконадежных серверных приложений
- J2ME – Java2 Micro Edition, упрощенная J2SE для применения в небольших устройствах с ограниченными аппаратными ресурсами

1-15. Какая версия Java поддерживается в большинстве браузеров? Что такое Java Plug-in?

a.) Большинство браузеров поддерживает уже устаревшую версию Java 1.1, хотя практически каждый имеет некоторые отличительные особенности, отклонения от спецификации и т.п. Это делает довольно трудоемким создание универсальных апплетов, предназначенных для всех пользователей Интернет.

Поэтому компания Sun с конца 1997 года предлагает специальный продукт Java Plug-in, который можно установить на любой браузер и который позволяет запускать апплеты в точном соответствии со спецификацией Java. Plug-in доступен для любой версии платформы.

1-16. Что такое JDK и JRE? В чем сходство и разница между ними? Какие основные утилиты входят в их состав?

a.) JDK – это Java Development Kit, средство разработчика Java, включающее в себя набор утилит, стандартные библиотеки с их сходным кодом и набор демонстрационных примеров. Утилиты включают в себя:

- java – реализация JVM
- javac – компилятор Java
- appletviewer – средство для запуска апплетов

- jar – архиватор формата JAR
- javadoc – утилита для автоматической генерации документации

JRE – это Java Runtime Environment, среда выполнения Java, предназначена только для запуска готовых Java-приложений, а потому содержит лишь реализацию виртуальной машины и набор стандартных библиотек.