УО «Полоцкий государственный университет»

Факультет информационных технологий

МЕТОДИЧЕСКИЕ УКАЗАНИЯ

к выполнению лабораторного практикума
по курсу «Технология разработки интерфейса программных средств»
для специальности
1-40 01 01 Программное обеспечение информационных технологий

Новополоцк 2011

Методические указания разработали:

Бураченок Ирина Брониславовна – ст. преподаватель кафедры технологий программирования,
Бураченок Алексей Леонидович – ст. преподаватель кафедры технологий программирования.

ЛАБОРАТОРНАЯ РАБОТА №1

ТЕМА: Организация диалога пользователя с программной системой при директивно-диалоговой форме взаимодействия.

ЦЕЛЬ: Изучение и приобретение навыков разработки интерфейса CLI (Command line interface) – директивно-диалоговых форм взаимодействия с программной системой на основе командных файлов (bat-файлов).

Результат обучения:

После успешного завершения занятия пользователь должен:

- Освоить методику разработки диалога типа CLI.
- Уметь программировать CLI.
- Знать основные команды MS DOS и их параметры.
- Знать методы создания bat-файлов.
- Уметь создавать командные файлы (bat-файлы) и их помощью осуществлять диалог пользователя с системой: с параметрами символами; с параметрами, использующие имена файлов; с использованием внешней команды "CHOICE"; с использованием нескольких параметров.

Используемая программа: операционная система Windows 2000 и выше, любой текстовый редактор.

План занятия:

1. Краткие теоретические сведения
2. Выполнение задания
3. Оформление отчета
4. Демонстрация примеров.

1. КРАТКИЕ ТЕОРЕТИЧЕСКИЕ СВЕДЕНИЯ

CLI (англ. *Command line interface*) – интерфейс командной строки — разновидность текстового интерфейса (CUI) между человеком и компьютером, в котором инструкции компьютеру даются в основном путём ввода с клавиатуры текстовых строк (команд), в UNIX-системах возможно применение мыши. Также известен под названием ***консоль***.

Основное назначение консоли. На устройстве-консоли, которое печатало текст на бумаге, интерфейс командной строки был единственным возможным. На видеотерминалах интерфейс командной строки применяется по таким причинам:

- небольшой расход памяти по сравнению с системой меню.
- в современном программном обеспечении имеется большое число команд, многие из которых нужны крайне редко. Поэтому даже в некоторых программах с графическим интерфейсом применяется командная строка: набор команды (при условии, что пользователь знает эту команду) осуществляется гораздо быстрее, чем, например, навигация по меню.

- естественное расширение интерфейса командной строки — пакетный интерфейс. Его суть в том, что в файл обычного текстового формата записывается последовательность команд, после чего этот файл можно выполнить в программе, что возымеет такой же (не меньший) эффект, как если бы эти команды были по очереди введены в командную строку.

Примеры — .bat-файлы в DOS и Windows, shell-скрипты в Unix-системах.

Если программа полностью или почти полностью может управляться командами интерфейса командной строки и поддерживает пакетный интерфейс, умелое сочетание интерфейса командной строки с графическим предоставляет пользователю очень мощные возможности.

Формат команды. Наиболее общий формат команд (в квадратные скобки помещены необязательные части):

`[символ_начала_команды]имя_команды [параметр_1 [параметр_2 […]]]`

Символ начала команды может быть самым разным, однако чаще всего для этой цели используется косая черта (/). Если строка вводится без этого символа, выполняется некоторая базовая команда: например, строка «Привет» в IRC эквивалентна вводу «`/msg Привет`». Если же такой базовой команды нет, символ начала команды отсутствует вообще (как, например, в DOS).

Параметры команд могут иметь самый разный формат. В основном применяются следующие правила:

– параметры разделяются пробелами (и отделяются от названия команды пробелом)

– параметры, содержащие пробелы, обрамляются кавычками-апострофами (') или двойными кавычками (")

– если параметр используется для обозначения включения какой-либо опции, выключенной по умолчанию, он начинается с косой черты (/) или дефиса (-)

– если параметр используется для включения/выключения какой-либо опции, он начинается (или заканчивается) знаком плюс или минус (для включения и выключения соответственно)

– если параметр указывает действие из группы действий, назначенных команде, он не начинается со специальных символов

– если параметр указывает объект, к которому применяется действие команды, он не начинается со специальных символов

– если параметр указывает дополнительный параметр какой-либо опции, то он имеет формат `/опция:дополнительный_параметр` (вместо косой черты также может употребляться дефис)

Например, в некоей абстрактной игре может быть такая команда:

`/map dm1 /skill:2`

Где

`/` — символ начала команды

`map` — название команды (переход на другой уровень)

`dm1` — обязательный параметр (название уровня)

`/skill:2` — дополнительный параметр (задание уровня сложности)

Применение. Основные сферы применения интерфейса командной строки: операционные системы, чаты, компьютерные игры.

CLI – директивно-диалоговая форма как правило предназначена для подготовленного пользователя и требует знания как алгоритмов выполнения программ,

так и отдельных команд и их параметров по управлению этими программами. Запуск программ или выполнение отдельных директив проводится с командной строки.

В диалоговом взаимодействии пользователя с программной системой выделяются 2 типа сообщений:

- входные сообщения, порождаемые пользователем с помощью средств ввода информации;
- выходные сообщения, формируемые системой с помощью средств вывода и отображения информации.

Первый шаг диалога чаще всего начинается с выдачи системой одного или нескольких выходных сообщений. Выходные сообщения, как правило, отражают результаты выполнения процедурной части, либо состояние системы и диалога. Последовательности диалога в свою очередь, делятся на последовательности, где инициатива может принадлежать системе и пользователю. Существует также и третий тип инициативы – смешанная инициатива, предполагающая периодическое перераспределение инициативы с помощью управляющих сигналов. Директивная форма взаимодействия требует определенных знаний системы и управляющих команд операционной системы.

Основные преимущества директивно-диалоговой формы взаимодействия с программной системой:

- Любую команду можно вызвать небольшим количеством нажатий.
- Можно обращаться к командам для разных исполнимых файлов почти мгновенно и непосредственно, тогда как в GUI приходится сначала запускать, а затем закрывать графический интерфейс для каждого исполнимого файла.
- Shell script в UNIX-подобных системах является полноценным интерпретируемым языком программирования и способен автоматизировать любую системную задачу. В Windows присутствует их примитивный аналог — пакетные файлы, по сути это, простейшая программируемость.
- Можно управлять программами, не имеющими графического интерфейса (например, выделенным сервером).
- Просмотрев содержимое консоли, можно повторно увидеть промелькнувшее сообщение, которое вы не успели прочитать.
- Можно пользоваться удаленным компьютером с любого устройства подключаемого к Интернету или локальной сети (ПК, субноутбук, КПК, сотовый телефон, портативная игровая консоль) без особых затрат трафика (единицы килобайт за сеанс).
- Отсутствие деталей интерфейса, таких как пусковые панели и рамки окон, что при равных разрешениях позволяет вместить значительно больше текста на страницу.

Недостатки директивно-диалоговой формы взаимодействия с программной системой:

- Интерфейс командной строки не является дружественным для пользователей, которые начали знакомство с компьютером с графического режима.
- Без автодополнения, ввод длинных и содержащих спецсимволы параметров с клавиатуры может быть затруднительным.
- Отсутствие «аналогового» ввода. Например подбор громкости с помощью озвученного ползунка позволяет выставить подходящую громкость быстрее, чем командой вроде `aumix -v 90`. (Однако, озвученный ползунок вполне может быть псевдографическим, что было выполнено в большинстве консольных плееров).

2. ВЫПОЛНЕНИЕ ЗАДАНИЯ

Задание к работе:

Необходимо вспомнить основные команды MS DOS и их параметры, а также методы создания bat-файлов.

Предварительно необходимо подготовить сценарий взаимодействия пользователя с программой (программами) на основе применения bat-файлов.

Выполнение данной работы состоит в создании четырех командных файлов, реализующих директивно-диалоговую форму взаимодействия пользователя с программной системой.

При выполнении работы необходимо создать командные файлы (bat-файлы), с помощью которых производится следующие типы диалогов пользователя с системой с использованием:

- параметров-символов;
- параметров, использующих имена файлов;
- внешней команды "CHOICE";
- нескольких параметров.

Количество используемых параметров и функций, исполняемые командными файлами, выбираются самим обучающимся.

3. ОФОРМЛЕНИЕ ОТЧЕТА

Содержание отчета:

1. Привести краткие сведения о формах диалогового взаимодействия.
2. Привести основные сведения о командных файлах и их практическом использовании для организации диалога пользователя с программной системой.
3. Привести тексты созданных bat-файлов.

Примечание:

1. Отчет должен быть представлен в соответствии с требованиями принятыми в ВУЗе.
2. Коды программ должны быть оригинальными у каждого студента.

Литература: 1 осн. [20-26], 2 осн. [80-98], 1 доп. [40-41], 7доп. [100-117], 9 доп. [162-234].

Продемонстрируйте Вашу работу преподавателю!

КОНТРОЛЬНЫЕ ВОПРОСЫ:

1. Как используются командные файлы при организации пользовательского интерфейса?
2. Какие параметры могут быть у bat-файлов?
3. Какова технология создания bat-файлов?

ЛАБОРАТОРНАЯ РАБОТА №2

ТЕМА: Организация диалога пользователя при использовании графического пользовательского интерфейса типа "Hand User Interface".

ЦЕЛЬ: Изучение и приобретение навыков разработки пользовательского интерфейса GUI с использованием гиперссылок в формате HTML.

Результат обучения:

После успешного завершения занятия пользователь должен:

- Освоить методику разработки диалога типа GUI.
- Уметь программировать GUI.
- Уметь создавать гиперссылки в формате HTML.

Используемая программа: Блокнот и Web-браузеры: Opera или Internet Explorer

План занятия:

1. Краткие теоретические сведения
2. Выполнение задания
3. Оформление отчета

1. КРАТКИЕ ТЕОРЕТИЧЕСКИЕ СВЕДЕНИЯ

GUI (англ. *Graphical user interface*) – графический пользовательский интерфейс (ГИП) — разновидность пользовательского интерфейса, в котором элементы интерфейса (меню, кнопки, значки, списки и т. п.), представленные пользователю на дисплее, исполнены в виде графических изображений.

Впервые концепция ГИП была предложена учеными из исследовательской лаборатории Xerox PARC в 1970-х.

В 1973 году в лаборатории Xerox PARC собрали молодых учёных и дали свободу исследований. В результате, кроме всего прочего, на свет появляется концепция графического интерфейса WIMP (Windows, Icons, Menus, Point-n-Click). В рамках этой концепции создаётся компьютер Alto.

В 1979 году Three Rivers Computer Corporation выпускает рабочую станцию PERQ, похожую по принципам построения на Alto. В 1981 году Xerox выпускает продолжение Alto — Star.

Коммерческое воплощение концепция ГИП получила в продуктах корпорации Apple Computer. В операционной системе AmigaOS ГИП с многозадачностью был использован в 1985 году. В настоящее время ГИП является стандартной составляющей большинства доступных на рынке операционных систем и приложений.

Основное назначение.

Примеры систем, использующих ГИП: Mac OS, Solaris, GNU/Linux, Microsoft Windows, NeXTSTEP, OS/2, BeOS. Однако в «чистом виде» данный вид пользовательского интерфейса используется в основном для карманных компьютеров, для которых характерен дисплей небольшого размера. В таких системах применяются объекты пользовательского интерфейса, учитывающие эту особенность.

Сегодня широко известны два основных класса PDA (Personal Digital Assistant-персональный цифровой ассистент – "карманный" компьютер, предназначенный, выполнения некоторых специальных функций) – в одних используется настоящий GUI-стиль как по внешнему виду, так и по поведению, в других применяется подмножество GUI-интерфейса. Для ввода данных пользователем применяется "жестикуляционный" стиль с пером и сенсорным экраном. Для поддержки PDA обычно используется GUI-ориентированное ПО для портативных или настольных компьютеров. Общий стиль для HUI-интерфейса можно назвать SIMP–стилем (Screen – экран, Icon – пиктограмма, Menu – меню и Pointer – указатель).

Выделяют следующие виды ГИП:

- простой: типовые экранные формы и стандартные элементы интерфейса, обеспечиваемые самой подсистемой ГИП;
- истинно-графический, двумерный: нестандартные элементы интерфейса и оригинальные метафоры, реализованные собственными средствами приложения или сторонней библиотекой;
- трёхмерный: на данный момент слабо классифицирован.

Достоинства. В отличие от интерфейса командной строки, в ГИП пользователь имеет произвольный доступ (с помощью устройств ввода – клавиатуры, мыши, джойстика и т. п.) ко всем видимым экранным объектам (элементам интерфейса) и осуществляет непосредственное манипулирование ими. Чаще всего элементы интерфейса в ГИП реализованы на основе метафор и отображают их назначение и свойства, что облегчает понимание и освоение программ неподготовленными пользователями.

Одним из требований к хорошему графическому интерфейсу программной системы является концепция «делай то, что я имею в виду» или DWIM (англ. Do What I Mean). DWIM требует, чтобы система работала предсказуемо, чтобы пользователь заранее интуитивно понимал, какое действие выполнит программа после получения его команды.

Наиболее распространённым способом построения ГИП является использование гиперссылки — фрагмента HTML (англ. *HyperText Markup Language* — «язык разметки гипертекста») и его базового элемента: указывающего на другой файл, который может быть расположен в Интернет (или содержащего полный путь (URL – (англ. *Uniform Resource Locator*) – единый указатель ресурсов) к этому файлу).

Гиперссылка для пользователя — это графическое изображение или текст на сайте, в письме электронной почты или в каком-либо электронном документе, устанавливающие связь и позволяющие переходить к другим объектам Интернет.

Для определения ссылки в HTML используется тег `<a>`, структура которого имеет вид `<a href="filename"> Текст ссылки </a>`, где «`filename`» — имя файла или адрес в Интернете, на который необходимо сослаться, а «`Текст ссылки`» — текст гипертекстовой ссылки, который будет непосредственно показан в HTML-документе.

Например, гипертекстовая ссылка:

`<a href="my_work.html"> Мои работы </a>` — ссылается на документ my_work.html, образуя гипертекстовую ссылку в виде слова «Мои работы»;

`<a href="photo/my_photo.html"> Мои фото </a>` — ссылается на файл my_photo.html, расположенный в каталоге photo, и образует ссылку в виде текста «Мой фотоальбом»;

`<a href="http://www.site.com">` — ссылается ресурс, расположенный на удаленном сервере.

2. ВЫПОЛНЕНИЕ ЗАДАНИЯ

Задание к работе:

Эта работа посвящена отработке навыков создания пользовательского интерфейса с использованием гиперссылок в формате HTML. Гиперссылки являются компонентами пользовательского интерфейса, учитывающие требования к проектированию "Hand User Interface".

Для освоения темы необходимо знать операторы и операторные конструкции, формирующие элементы GUI (например, на VBA).

Знать методы использования элементов GUI для организации диалоговой надстройки.

Предварительно необходимо подготовить сценарий взаимодействия пользователя с прикладной программой (программами) на основе GUI.

Спроектировать и реализовать логически организованную группу гипертекстовых документов (HTML-файлов), систему ссылок внутри документов, представляющими пользователю удобную и интуитивно понятную навигацию по гипертексту.

Примечание:

В отчете, кроме обязательных компонент, должна быть представлена навигационная схема по группе гипертекстовых документов (HTML-файлов).

Для отладки проекта используется технология "петли" – (loop back), браузер и сервер размещаются на одном компьютере. Все созданные файлы должны быть размещены на локальном WEB-сервере.

3. ОФОРМЛЕНИЕ ОТЧЕТА

Содержание отчета:

1. Привести краткие сведения об интерфейсах типа "Hand User Interface".
2. Привести основные сведения о технологии "петли" – (loopback)
3. Привести тексты созданных гипертекстовых документов (HTML-файлов).

Примечание:

1. Отчет должен быть представлен в соответствии с требованиями принятыми в ВУЗе.
2. Тексты, созданных гипертекстовых документов (HTML-файлов) должны быть индивидуальными у каждого студента.

Литература: 1 осн. [20-26], 2 осн. [80-98], 7 доп. [100-117], 9 доп [162-234].

Продемонстрируйте Вашу работу преподавателю!

КОНТРОЛЬНЫЕ ВОПРОСЫ:

1. Какие компоненты пользовательского интерфейса используются в интерфейсе типа "Hand User Interface"?
2. Какова технология создания гиперссылок?
3. Как проектируется структура диалогового взаимодействия в виде навигационной схемы?

ЛАБОРАТОРНАЯ РАБОТА №3

ТЕМА: Организация диалога пользователя при использовании пользовательского интерфейса на основе "Web User Interface".

ЦЕЛЬ: Изучение и приобретение навыков разработки пользовательского интерфейса WUI.

Результат обучения:

После успешного завершения занятия пользователь должен:

- Уметь описывать структуру диалогового взаимодействия в виде навигационной схемы.
- Освоить методику разработки диалогового типа WUI.
- Уметь программировать WUI.
- Уметь создавать HTML-страницы.

Используемая программа: Блокнот и Web-браузеры: Opera или Internet Explorer

План занятия:

1. Краткие теоретические сведения
2. Выполнение задания
3. Оформление отчета

1. КРАТКИЕ ТЕОРЕТИЧЕСКИЕ СВЕДЕНИЯ

Web-приложение — клиент-серверное приложение, в котором клиентом выступает браузер, а сервером — веб-сервер. Логика Web-приложения распределена между сервером и клиентом, хранение данных осуществляется, преимущественно, на сервере, обмен информацией происходит по сети. Одним из преимуществ такого подхода является тот факт, что клиенты не зависят от конкретной операционной системы пользователя, поэтому Web-приложения являются межплатформенными сервисами.

Web-приложения стали широко популярными в конце 1990-х — начале 2000-х годов.

Существенное преимущество построения Web приложений для поддержки стандартных функций браузера заключается в том, что функции должны выполняться независимо от операционной системы данного клиента. Вместо того чтобы писать различные версии для Microsoft Windows, Mac OS X, GNU/Linux и других операционных систем, приложение создается один раз для произвольно выбранной платформы и на ней разворачивается. Однако различная реализация HTML, CSS, DOM и других спецификаций в браузерах может вызвать проблемы при разработке Web-приложений и последующей поддержке. Кроме того, возможность пользователя настраивать многие параметры браузера (например, размер шрифта, цвета, отключение поддержки сценариев) может препятствовать корректной работе приложения.

Другой (менее универсальный) подход заключается в использовании Adobe Flash, Silverlight или Java-апплетов для полной или частичной реализации пользовательского интерфейса. Поскольку большинство браузеров поддерживает эти технологии (как правило, с помощью плагинов), Flash- или Java-приложения могут

выполняться с легкостью. Так как они предоставляют программисту больший контроль над интерфейсом, они способны обходить многие несовместимости в конфигурациях браузеров, хотя несовместимость между Java или Flash реализациями на стороне клиента может приводить к различным осложнениям.

Web-приложение состоит из клиентской и серверной частей, тем самым реализуя технологию «клиент-сервер». Клиентская часть реализует пользовательский интерфейс, формирует запросы к серверу и обрабатывает ответы от него. Серверная часть получает запрос от клиента, выполняет вычисления, после этого формирует веб-страницу и отправляет её клиенту по сети с использованием протокола HTTP.

Само Web-приложение может выступать в качестве клиента других служб, например, базы данных или другого Web-приложения, расположенного на другом сервере. Ярким примером Web-приложения является система управления содержимым статей Википедии: множество её участников могут принимать участие в создании сетевой энциклопедии, используя для этого браузеры своих операционных систем (будь то Microsoft Windows, GNU/Linux или любая другая операционная система) и не загружая дополнительных исполняемых модулей для работы с базой данных статей.

В настоящее время набирает популярность новый подход к разработке Web-приложений, называемый Ajax. При использовании Ajax страницы Web-приложения не перезагружаются целиком, а лишь догружают необходимые данные с сервера, что делает их более интерактивными и производительными.

Для создания Web-приложений на стороне сервера используются разнообразные технологии и любые языки программирования, способные осуществлять вывод в стандартную консоль, например,

Название	**Веб-сервер**
ASP	специализированный
ASP.NET	специализированный
C/C++	практически любой
Java	множество, в том числе свободных
Perl	практически любой
PHP	практически любой

На стороне клиента используется:

Для реализации GUI: HTML, CSS

Для формирования и обработки запросов, создания интерактивного и независимого от браузера интерфейса: ActiveX, Adobe Flash, Adobe Flex, Java, JavaScript, Silverlight

Компоненты пользовательского интерфейса на основе WUI обеспечивают взаимодействие пользователя в сетевых программных приложениях (например, в Интернет). К одной из важнейших функций Web-страниц (при наличии объектов WUI), помимо непосредственного отображения информации для пользователей, относится возможность посылать на Web-узел определенные данные и производить их обработку на сервере. Для этих целей в код страницы включаются специальные тэги, определяющие в HTML-странице специальные объекты-формы, с помощью которых можно создавать интерактивный интерфейс.

2. ВЫПОЛНЕНИЕ ЗАДАНИЯ

Задание к работе:

Для освоения темы необходимо знать операторы и операторные конструкции, формирующие элементы WUI (например, на HTML).

Знать методы использования элементов WUI для организации диалога.

Предварительно необходимо подготовить сценарий взаимодействия пользователя с внешней программой (программами) на основе WUI.

Отработать навыки создания в HTML-документе компонентов, позволяющих создавать интерактивное взаимодействие пользователя с WWW-сервером (HTTP – сервер).

Составить код HTML-страницы с включением тегов FORM со всеми компонентами, которые формируют интерактивный интерфейс пользователя с программным приложением.

Составить код HTML-страницы с определенной смысловой нагрузкой, выбрать компоненты форм, определяющие вводимую информацию.

Примечание:

Реакцию сервера в этой работе следует проверять на основе оценки содержимого переменных окружения сервера. Для этого использовать серверную скрипт-программу, например на языке PERL.

Для отладки Web-проектов используется технология "петли" – (loopback), браузер и сервер размещаются на одном компьютере.

3. ОФОРМЛЕНИЕ ОТЧЕТА

Содержание отчета:

1. Привести краткие сведения об интерфейсах типа " Web User Interface ".
2. Привести основные сведения о технологии "петли" – (loopback)
3. Привести код созданной HTML-страницы.

Примечание:

1. Отчет должен быть представлен в соответствии с требованиями принятыми в ВУЗе.
2. Код, созданной HTML-страницы должен быть индивидуальными у каждого студента.

Литература: 1 осн. [20-26], 2 осн. [80-98], 7 доп. [100-117], 9 доп. [162-234].

Продемонстрируйте Вашу работу преподавателю!

КОНТРОЛЬНЫЕ ВОПРОСЫ:

1. Какие компоненты пользовательского интерфейса используются в интерфейсе типа "Web User Interface"?
2. Какова технология создания компонент пользовательского интерфейса?
3. Как программируются компоненты, позволяющие посылать данные на сервер?

ЛАБОРАТОРНАЯ РАБОТА №4

ТЕМА: Выбор компонентов WUI (Web User Interface) для организации диалога пользователя в зависимости от решаемых задач.

ЦЕЛЬ: Изучение и приобретение навыков разработки дружественного пользовательского интерфейса WUI.

Результат обучения:

После успешного завершения занятия пользователь должен:

- Понять смысл "дружественного" интерфейса пользовательского интерфейса.
- Уметь создавать "дружественные" интерфейсы.
- Освоить методику разработки диалогового типа WUI.
- Уметь программировать WUI.
- Уметь осуществлять выбор компонентов WUI при создании дружественного интерфейса.

Используемая программа: Блокнот и Web-браузеры: Opera или Internet Explorer

План занятия:

1. Краткие теоретические сведения
2. Выполнение задания
3. Оформление отчета

1. КРАТКИЕ ТЕОРЕТИЧЕСКИЕ СВЕДЕНИЯ

Используемые визуальные компоненты при создании Web-интерфейсов:

Управление общей компоновкой обеспечивают компоненты группы Layout, которые позволяют разметить блоками всю область окна приложения и в дальнейшем работать с ними по отдельности, обеспечивая в то же время общую целостность интерфейса и его структуры. Отдельные части страницы можно сворачивать, скрывая полностью или менять их размер, передвигать границы областей – все, что привыкли делать пользователи в обычных приложениях.

Динамическая подсказка или Tooltip – позволяет для любого элемента на странице (как визуального, так и к любому DOM-объекту) прицепить подсказку, которая будет показана при наведении мыши. Сама подсказка может содержать в себе произвольный HTML-код, изображения и даже быть контекстно-зависимой – подгружаться через AJAX.

При необходимости можно подключать и использовать множество *виджетов* – кнопки (разного вида, с изображениями и текстом, кнопки-переключатели, кнопки с выпадающим меню и т.п.), *меню* (как контекстные, так и тулбары с меню, совсем как в обычных приложениях), *индикаторы загрузки* и *прогресс-бары*. Есть и такое, свойственное только Web-приложениям решение, как *Loading Mask*, которое позволяет "заморозить" для пользователя все приложение на то время, пока получаются и обновляются данные с сервера. Есть и *функциональные компоненты для выбора даты* и *цвета*, есть компоненты *для вывода сообщений*, как модальные MessageBox, так и

полноценные окна, в которые, в свою очередь, можно вставлять любые элементы управления или разделять их закладки, сворачивать и развертывать, перетаскивать по экрану и т.п.

Таб-панель – еще один мощный компонент для построения тех приложений, которые, с одной стороны, требуют, чтобы пользователю показывалось сразу много разной информации, формы, таблицы и другие объекты, в то же время нужны ограничения, и в каждый момент времени показывать только самое необходимое. Для такого структурирования широко применяют вкладочный интерфейс.

Tree – часто используемый компонент, позволяющий группировать иерархические структуры и отображать их с любой глубиной вложенности, автоматически подгружая новые элементы, менять местами, перетаскивая мышью отдельные элементы и другие стандартные операции.

Форма – является одной из основных высокоуровневых абстракций библиотеки, и использует почти все из описанных компонентов, в частности большинство визуальных виджетов, которые расширяют стандартные элементы форм. Она также использует слой доступа к данным для заполнения некоторых компонентов (например, ComboBox может получать данные от сервера). В функциональность форм входит и предварительная проверка вводимых пользователем значений, что уже стало классической функциональностью для Веб. Используя встроенные обработчики, можно проверять вводимые e-mail адреса, URL и просто строки символов, однако никто не мешает расширять встроенные наборы, создавая пользовательские валидаторы (любой виджет для ввода данных позволяет определить произвольную функцию для проверки ввода). Есть и встроенный HTML-редактор.

Таблица Grid – самый востребованный и самый мощный из компонентов, которым располагает разработчик. Впрочем, он одновременно является и самым сложным. В общих чертах, он также построен на модели абстракции каждого уровня: данные могут получаться из DataStore, внешнее отображение задается как специальными функциями-рендерами для каждого столбца отдельно, так и компонентом GridView, который отвечает, например, за заголовок таблицы и создание тулбара с элементами постраничного управления выводом данных. Для выделения строк и ячеек есть, соответственно, RowSelectinsModel и CellSelectionModel. Также поддерживается особый тип таблиц со встроенной функцией редактирования данные напрямую в ячейке (это позволяет расширить функционал такой таблицы почти до уровня Excel/Calc).

2. ВЫПОЛНЕНИЕ ЗАДАНИЯ

Задание к работе:

Необходимо изучить сведения о способах создания "дружественного" пользовательского интерфейса. Знать программный инструментарий, позволяющий создавать "дружественный" интерфейс (например, для WUI Java-script).

В данной работе применяются специальные возможности программного инструментария из группы Web-программирования, например, Java-script или VB-script. Используются возможности "событий", реализующихся как реакция программной системы на действия пользователя.

"События" при правильном применении могут оказать эффективную помощь пользователю, например: при вводе и отправке данных, подсказкой и другими эффектами.

Предварительно необходимо проанализировать структуру диалогового взаимодействия с внешней программой (программами) на сервере. На основе анализа выбрать и программно реализовать компоненты WUI.

Освоить навыки создания WUI с элементами помощи пользователю.

В работе предлагается разработать интерактивный интерфейс с проверкой его работоспособности с использованием HTTP-сервера на локальном компьютере. Запросы могут обрабатываются серверным CGI-сценарий в виде script-программы на языке PERL. PERL-программа размещается в специальной папке, находящейся на HTTP-сервере. Минимальный вариант выполнения работы в виде сайта с определенной смысловой нагрузкой и использованием "дружественного интерфейса" реализованного на скрипт языках исполняющихся на стороне клиента.

Примечание:

Для отладки Web-проектов используется технология "петли" – (loopback), браузер и сервер размещаются на одном компьютере.

3. ОФОРМЛЕНИЕ ОТЧЕТА

Содержание отчета:

1. Перечислить особенности использования различных компонентов при создании дружественного интерфейса типа "Web User Interface".
2. Привести основные сведения о технологии "петли" – (loopback)
3. Привести примеры script-программ с проверкой работоспособности интерактивного интерфейса.

Примечание:

1. Отчет должен быть представлен в соответствии с требованиями принятыми в ВУЗе.
2. Script-программы должны быть индивидуальными у каждого студента.

Литература: 1 осн. [20-26], 2 осн. [80-98], 7 доп. [100-117], 9 доп. [162-234].

Продемонстрируйте Вашу работу преподавателю!

КОНТРОЛЬНЫЕ ВОПРОСЫ:

1. Что означает термин "дружественный" интерфейс?
2. Как применяются "события" при создании пользовательского интерфейса?
3. Какова методика отладки по технологии loopback?

ЛАБОРАТОРНАЯ РАБОТА №5

ТЕМА: Комплексное решение вопросов создания интерфейсов на основе "Web User Interface" и "Common Gateway Interface".

ЦЕЛЬ: Изучение и приобретение навыков разработки пользовательского интерфейса на основе WUI-CGI.

Результат обучения:

После успешного завершения занятия пользователь должен:

- Освоить методику комплексной разработки на основе взаимодействия интерфейсов типа WUI-CGI.
- Уметь программировать WUI и CGI.

Используемая программа: Блокнот и Web-браузеры: Opera или Internet Explorer

План занятия:

1. Краткие теоретические сведения
2. Выполнение задания
3. Оформление отчета

1. КРАТКИЕ ТЕОРЕТИЧЕСКИЕ СВЕДЕНИЯ

CGI (от англ. Common Gateway Interface — «общий интерфейс шлюза») — стандарт интерфейса, используемого для связи внешней программы с веб-сервером. Программу, которая работает по такому интерфейсу совместно с веб-сервером, принято называть шлюзом, хотя многие предпочитают названия «скрипт» (сценарий) или «CGI-программа».

Сам интерфейс разработан таким образом, чтобы можно было использовать любой язык программирования, который может работать со стандартными устройствами ввода/вывода. Такими возможностями обладают даже скрипты для встроенных командных интерпретаторов операционных систем, поэтому в тех случаях, когда нет нужды в сложной функциональности, могут использоваться даже такие простые командные скрипты.

Все скрипты, как правило, помещают в каталог cgi (или cgi-bin) сервера, но это необязательно: скрипт может располагаться где угодно, но при этом большинство веб-серверов требуют специальной настройки. В веб-сервере Apache, например, такая настройка может производиться при помощи общего файла настроек httpd.conf или с помощью файла .htaccess в том каталоге, где содержится этот скрипт.

CGI является одним из наиболее распространённых средств создания динамических веб-страниц.

2. ВЫПОЛНЕНИЕ ЗАДАНИЯ

Задание к работе:

Освоить и закрепить навыки создания WUI-CGI интерфейсов.

В данной работе используются навыки и знания, полученные при выполнении предыдущих работ. Работа выполняется по вариантам.

Необходимо создать WUI интерфейс в виде HTML-страницы с количеством объектов согласно варианту.

В одном объекте необходима проверка вводимых данных.

Необходимо получить ответ сервера из всех объектов в FORM (по вариантам).

Необходимо часть данных записать в отдельный файл на стороне сервера (по вариантам).

Рекомендации: Полезно использовать коды из программ, выполненных в предыдущих лабораторных работах.

Примечание: Для отладки Web-проектов используется технология "петли" – (loopback), броузер и сервер размещаются на одном компьютере.

3. ОФОРМЛЕНИЕ ОТЧЕТА

Содержание отчета:

1. Привести краткие сведения об интерфейсах типа WUI (Web User Interface).
2. Привести краткие сведения об интерфейсах типа CGI (Common Gateway Interface).
3. Привести script-программы с проверкой работоспособности интерактивного интерфейса.

Примечание:

1. Отчет должен быть представлен в соответствии с требованиями принятыми в ВУЗе.
2. Script-программы должны быть индивидуальными у каждого студента.

Литература: 1 осн. [231-267], 5 осн. [161-194], 6 доп. [182-197], 9 доп. [211-324].

Продемонстрируйте Вашу работу преподавателю!

КОНТРОЛЬНЫЕ ВОПРОСЫ:

1. В чем смысл и различие понятий: программы, исполняющиеся на стороне клиента и программы, исполняющиеся на стороне клиента?
2. Как выбираются компоненты объекта FORM для реализации пользовательского интерфейса?
3. Как учитывается метод посылки данных во внешней программе?

ЛАБОРАТОРНАЯ РАБОТА №6

ТЕМА: Разработка пользовательского интерфейса на основе "Graphic User Interface".

ЦЕЛЬ: Изучение и приобретение навыков разработки пользовательского интерфейса на основе GUI.

Результат обучения:

После успешного завершения занятия пользователь должен:

- Освоить методику разработки GUI.
- Уметь программировать GUI.
- Уметь создавать макросы в MS Excel и MS Word.

Используемая программа: MS Excel и MS Word

План занятия:

1. Краткие теоретические сведения
2. Выполнение задания
3. Оформление отчета

1. КРАТКИЕ ТЕОРЕТИЧЕСКИЕ СВЕДЕНИЯ

Visual Basic for Applications (VBA, Visual Basic для приложений) — немного упрощённая реализация языка программирования Visual Basic, встроенная в линейку продуктов Microsoft Office (включая версии для Mac OS), а также во многие другие программные пакеты, такие как AutoCAD, SolidWorks, CorelDRAW, WordPerfect и ESRI ArcGIS. VBA покрывает и расширяет функциональность ранее использовавшихся специализированных макро-языков, таких как WordBasic.

VBA является интерпретируемым языком. Как и следует из его названия, VBA близок к Visual Basic. VBA, будучи языком, построенным на COM, позволяет использовать все доступные в операционной системе COM объекты и компоненты ActiveX. По сути, возможно создание приложения на основе Microsoft Word VBA, использующего только средства Corel Draw.

В будущем Microsoft планирует заменить VBA на Visual Studio Tools for Applications (VSTA) — инструментарий расширения функциональности приложений, основанный на Microsoft .NET.

К достоинствам языка можно отнести сравнительную лёгкость освоения, благодаря которой приложения могут создавать даже пользователи, не программирующие профессионально.

К недостаткам именно VBA, если не рассматривать недостатки Basic в целом, можно отнести невозможность создания более-менее автономного кода и слишком высокую открытость кода для случайного изменения.

2. ВЫПОЛНЕНИЕ ЗАДАНИЯ

Задание к работе:

Освоить навыки создания и использования в приложениях элементов пользовательского интерфейса на основе GUI (Graphic User Interface).

Разработать два приложения в MS Excel и в MS Word с элементами GUI.

Обязательными компонентами являются объекты DialogSheet или UserForm для приложения в VBA MS Excel и элементы из опции Формы панели инструментов MS Word.

Общая рекомендация: проанализируйте коды макросов.

Примечание: Работы каждого студента должны быть оригинальными.

3. ОФОРМЛЕНИЕ ОТЧЕТА

Содержание отчета:

1. Привести сведения об интерфейсах GUI.
2. Привести сведения о компонентах VBA используемых для создания пользовательского интерфейса GUI.
3. Привести примеры кодов макросов.

Примечание:

1. Отчет должен быть представлен в соответствии с требованиями принятыми в ВУЗе.
2. Коды макросов должны быть индивидуальными у каждого студента.

Литература: 4 осн. [105-193], 5 осн. [22-37], 1 доп. [15-29], 2 доп. [261-279].

Продемонстрируйте Вашу работу преподавателю!

КОНТРОЛЬНЫЕ ВОПРОСЫ:

1. В чем отличие интерфейсов WUI и GUI?
2. Какие компоненты VBA используются для создания пользовательского интерфейса GUI?
3. Какова технология создания макросов, реализующих пользовательский интерфейс GUI?

СПИСОК ЛИТЕРАТУРЫ

Основная литература:

1. Джеф Раскин, Интерфейс: новые направления в проектировании компьютерных систем. – Пер. с англ. - СПб: Символ-Плюс, 2003.

2. Торрес Роберт Дж., Практическое руководство по проектированию и разработке пользовательского интерфейса. – Пер. с англ. – М. Издательский дом "Вильямс", 2002.

3. Коутс Р., Влеймник И. Интерфейс "человек-машина " – Пер. с англ. –М.: Мир, 1990.

4. Алиев Т.М., Вигдоров Д.И., Кривошеев В.П. Системы отображения информации.//М.: Высшая школа ,1988.

5. Гасов В.М., Соломонов Л.А. Инженерно-психологическое проектирование взаимодействия человека с техническими средствами. Практическое пособие. Под ред. Четверикова В.Н. //М.: Высшая школа, 1990.

6. Соломонов Л.А., Филипович Ю.Н., Шульгин В.А. Персональные автоматизированные информационные системы дисплейные комплексы. Практическое пособие. Под ред. Четверикова В.Н. //М.: Высшая школа, 1990.

7. Гасов В.М., Меньков А.В., Соломонов Л.А., Шигин А.В. Системное проектирование взаимодействия человека с техническими системами. Практическое пособие. Под ред. Четверикова В.Н. //М.: Высшая школа, 1991.

8. Гасов В.М., Коротаев А.И., Сенькин С.И. Отображение информации. Практиеское пособие. Под ред. Четверикова В.Н. //М.: Высшая школа, 1991.

9. Сальников Ю.В., Савченко А.В., Филипов А.Н. Средства общения с ЭВМ. Под ред. Савельева А.Я. //М.: Высшая школа.,1987.

Дополнительная литература:

1. Анохин А.Н., Острейковский В.А. Организация взаимодействия человека с ЭВМ. Учебное пособие по курсу "Средства взаимодействия человека с ЭВМ".//Обнинск, ОИАТЭ, 1990.

2. Айден К., Колесниченко О., Крамер М., Фибельман Х., Шишигин И. Аппаратные средства PC.// Спб: BHV, 1998.

3. Борзенко А. IBM PC: устройство, ремонт, модернизация.//М.:1995. 450 с.

4. Венда В.Ф., Инженерная психология и синтез систем отображения информации. //Машиностроение, 1975.

5. Смоляров А.М. Системы отображения информации и инженерная психология. //М.: Высшая школа, 1982.

6. Дракин В.И., Попов Э.В., Преображенский А.Б. Общение конечных пользователей с системами обработки данных.//М.: Радио и связь, 1988,

7. Основы инженерной психологии. Под ред. В.Ф. Ломова. – М.: Высшая школа 1986.

8. Гультяев А.К., Машин В.А. Уроки WEB-мастера. Технология и инструменты: Практическое пособие. Спб.: КОРОНА, 2001.

9. Ганеев Р.М. Проектирование интерактивных WEB-приложений: учебное пособие – М.: Горячая линия-Телеком, 2001.