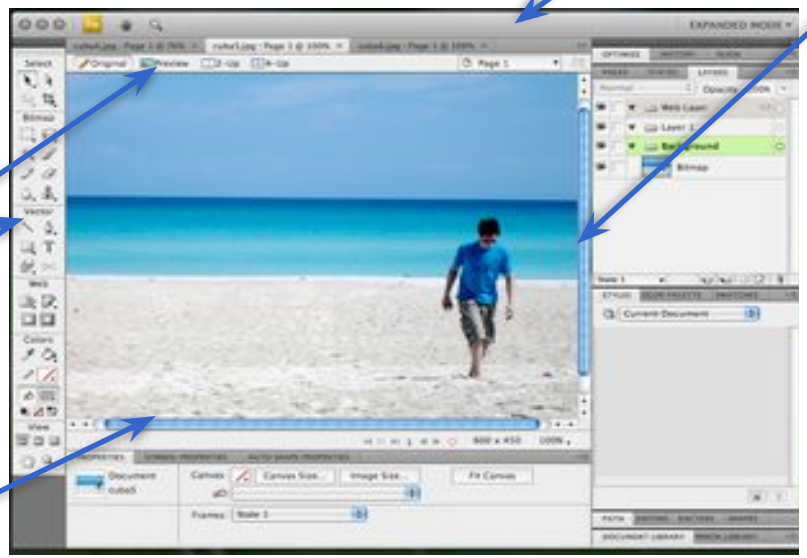
ЭЛЕМЕНТЫ ГРАФИЧЕСКОГО ИНТЕРФЕЙСА

Окно – выделенная область экрана, визуально разграничивающая одновременно выполняемые процессы.

Окно

- Главные окна программы
- Окна документа
- Диалоговые окна
- Палитры
- Окна браузера

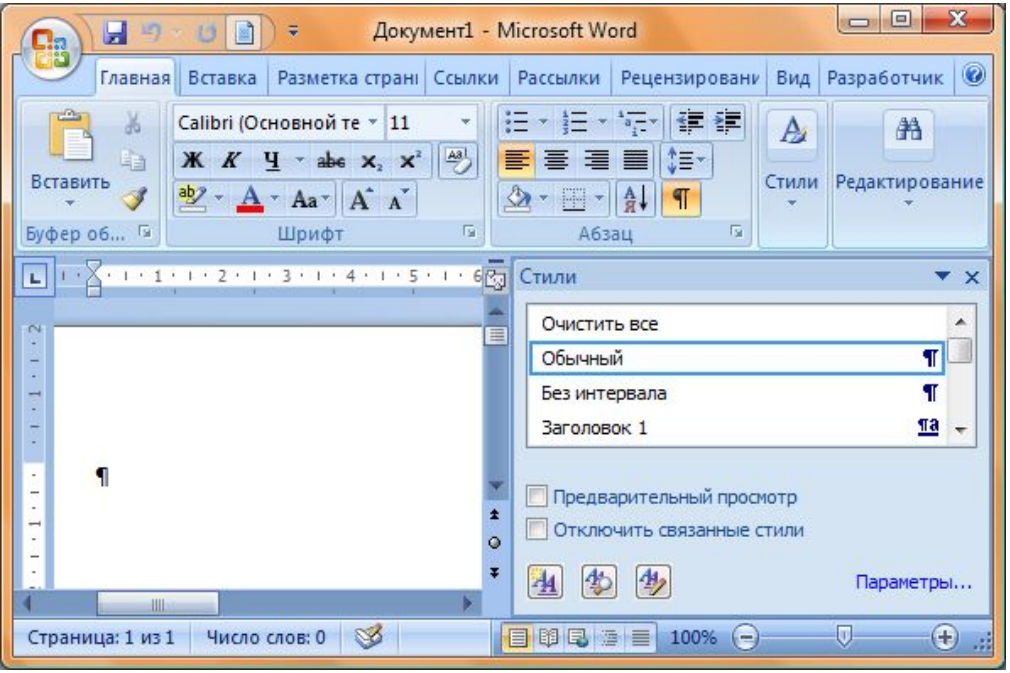
Элементы окна

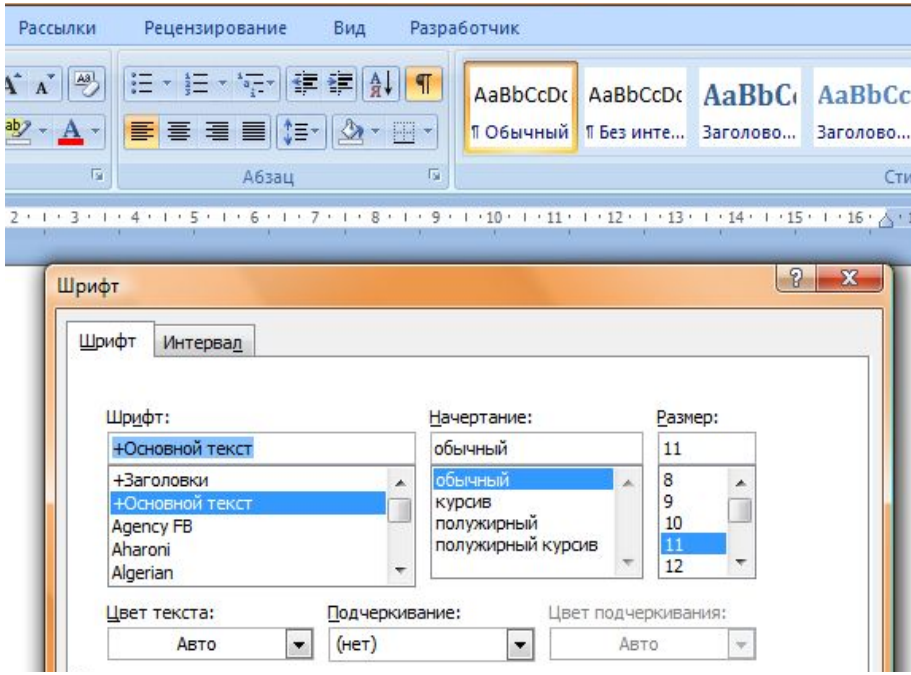


МОДАЛЬНЫЕ, НЕМОДАЛЬНЫЕ ОКНА

Немодальные (англ. modeless) диалоговые окна используются в случаях, когда выводимая в окне информация не является существенной для дальнейшей работы системы. Поэтому окно может оставаться открытым, в то время как работа пользователя с системой продолжается.

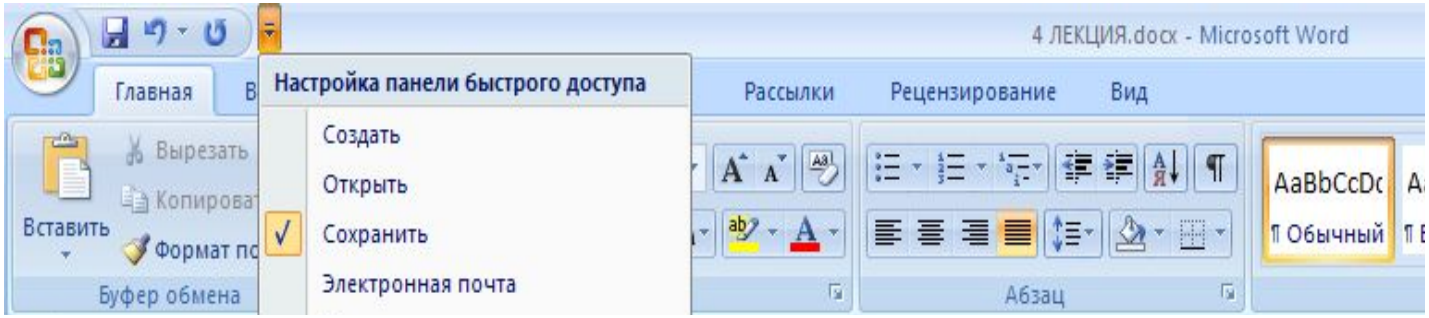
Модальным называется окно, которое блокирует работу пользователя с родительским приложением до тех пор, пока пользователь это окно не закроет. Диалоговые окна преимущественно реализованы модальными.





СТРОКА ЗАГОЛОВКА, СТРОКА СТАТУСА, ПОЛОСА ПРОКРУТКИ

Строка заголовка



Строку заголовка можно использовать для вызова функций, которые нужны только наиболее опытной аудитории: нажатие на пиктограмму в строке заголовка вызывает раскрывающееся меню.

Строка статуса предназначена для двух вещей: она может быть либо собственно строкой статуса, т.е. отображать текущее состояние системы, либо быть панелью инструментов для опытных пользователей (или же делать и то, и другое).

Полоса прокрутки – элемент окна, осуществляющий перелистывание его содержимого. Не очень удачный, с точки зрения удобства работы, элемент интерфейса. С появлением мыши со скрольным колесиком необходимость в полосе прокрутки значительно снизилась.

ПАНЕЛЬ ИНСТРУМЕНТОВ

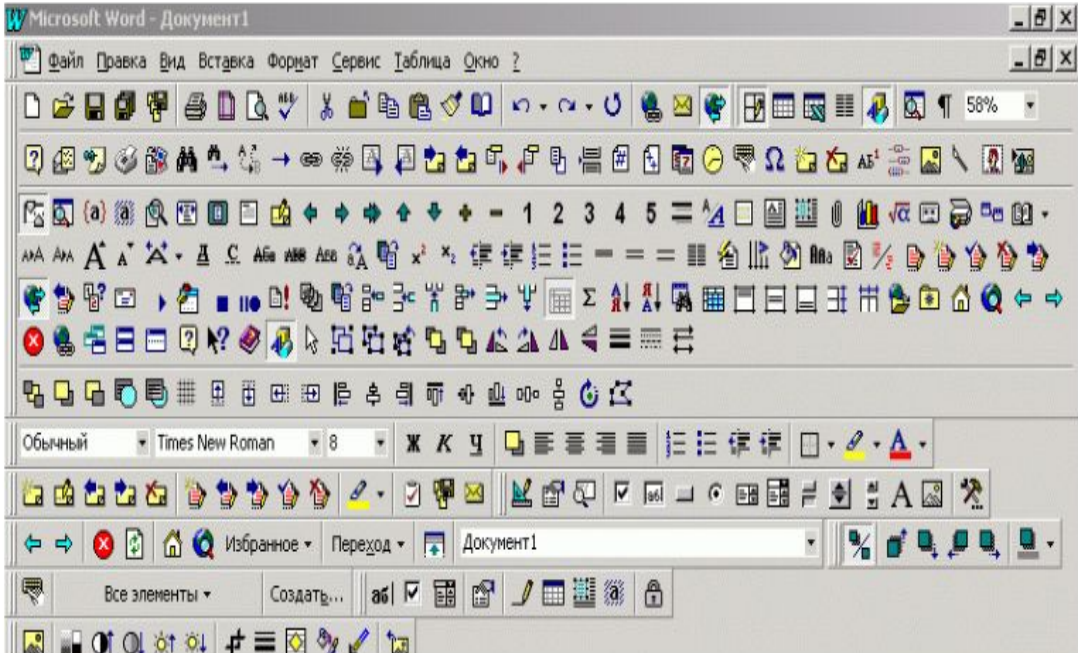
Панели инструментов представляют собой наборы пиктограмм, выбор которых инициирует какое-либо действие.

Все панели имеют следующие достоинства:

- ✔ они позволяют пользователям быстро вызывать нужные функции мышью
- ✔ они позволяют пользователям меньше задействовать память
- ✔ они повышают визуальное богатство интерфейса
- ✔ они ускоряют обучение работе с системой (по сравнению с раскрывающимся меню) благодаря своей большей наглядности.

Во-первых, можно (и нужно) помещать в панель только наиболее часто используемые команды (возможность индивидуальной настройки панели пользователем).

Во-вторых, панель можно сделать зависимой от контекста действий пользователя.



МЕНЮ

Меню – это метод взаимодействия пользователя с системой, при котором пользователь выбирает из предложенных вариантов, а не предоставляет системе свою команду. Соответственно, диалоговое окно с несколькими кнопками (и без единого поля ввода) также является меню.

Первая классификация делит меню на два типа:

- *Статические меню,* т.е. меню, постоянно присутствующие на экране. Характерным примером такого типа меню является панель инструментов.
- *Динамические меню,* в которых пользователь должен вызвать меню, чтобы выбрать какой либо элемент. Примером является обычное контекстное меню.

Вторая классификация также делит меню на два типа:

- Меню, разворачивающиеся в пространстве (например, обычное выпадающее меню).
- Меню, разворачивающееся во времени. При использовании таких меню элементы верхнего уровня по тем или иным причинам исчезают с экрана. Например, диалоговое окно с меню может перекрывать элемент управления, которым это меню было вызвано.

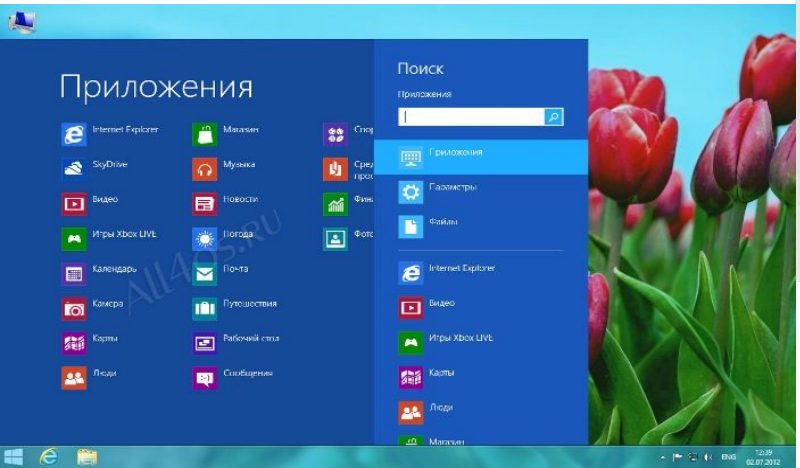
КНОПКИ

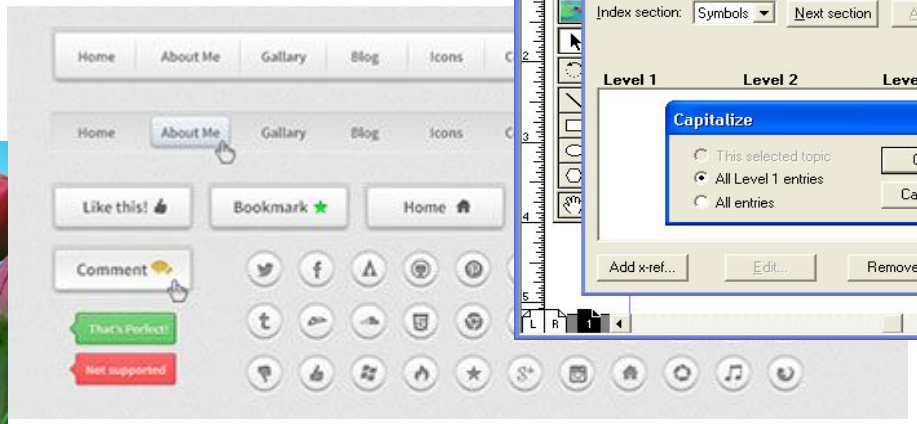
Кнопкой называется элемент управления, позволяющий выбрать опцию или вызвать событие (например, запуск подпрограммы). Все взаимодействие пользователя с кнопкой ограничивается нажатием.

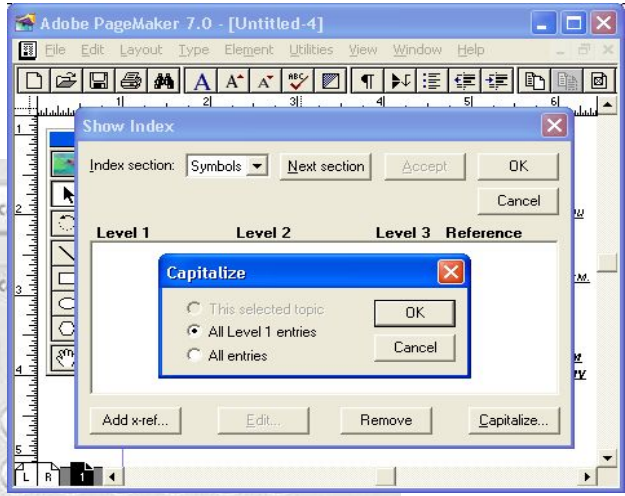
Командные кнопки (Push Buttons) – нажатие на такую кнопку запускает какое-либо явное действие (кнопки прямого действия). Изображаются в виде прямоугольника, в центре размещается короткое текстовое сообщение (слово), поясняющее, какое именно событие инициирует нажатие кнопки.

Кнопка имеет три состояния:

- нормальное (кнопка доступна),
- нажатое (активированное)
- недоступное (неактивное)







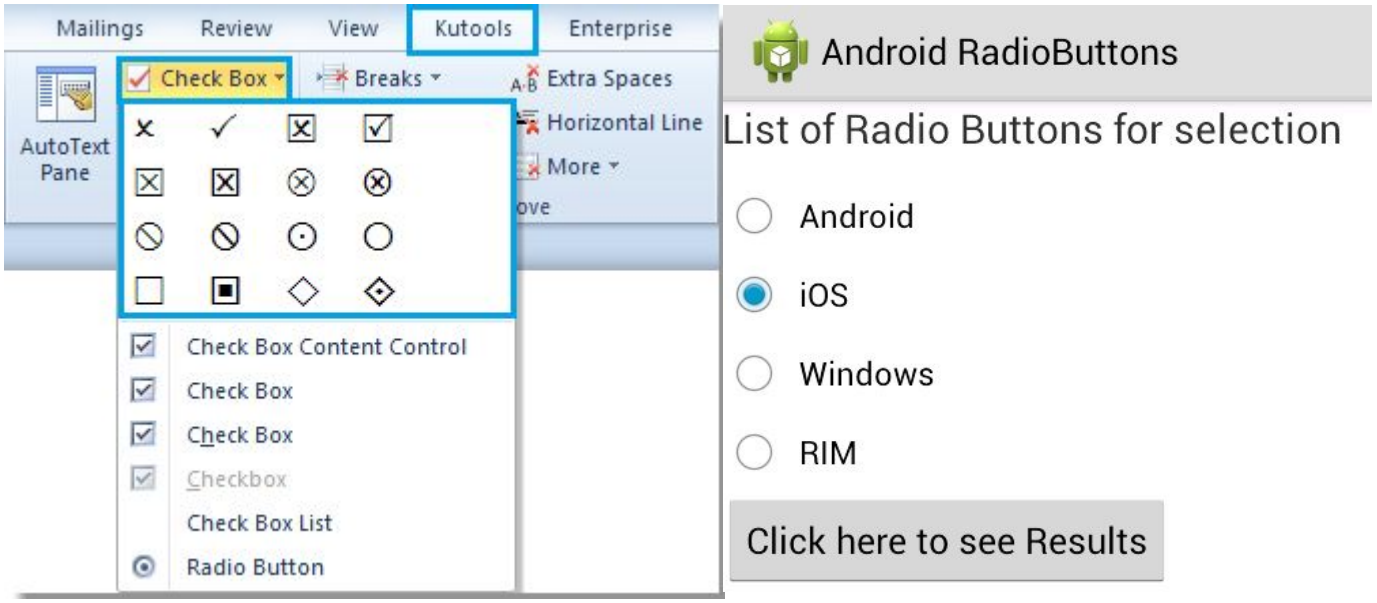
- ✔ **Командные кнопки**
- ✔ **Кнопки доступа к меню**

ПОЛЯ ВВОДА: ЧЕКБОКСЫ И РАДИОКНОПКИ

Чекбоксы (Checkboxes) и **радиокнопки** (Radio buttons) – кнопки отложенного действия, т.е. их нажатие не инициирует какое-либо немедленное действие, с их помощью пользователя вводят некоторые параметры, которые скажутся после, когда действие будет запущено другими элементами управления.

Главное отличие состоит в том, что:

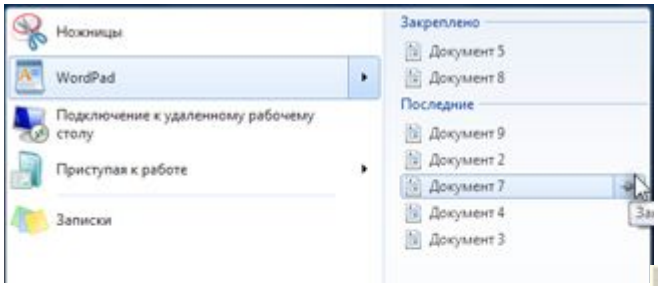
- радиокнопки являются кнопками единственного выбора
- чекбоксы – множественного.

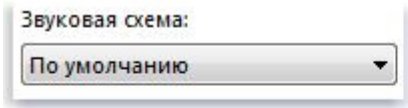


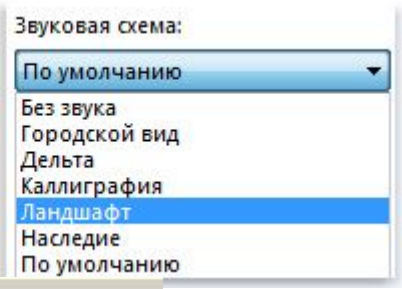
СПИСКИ

Вариантом реализации радиокнопок и чекбоксов являются **списки** (list boxes) – специализированные средства управления, которые отображают раскрывающиеся перечни значений и позволяют пользователю выбирать значение из списка, или вводить другое значение в присоединенное текстовое поле.

Списки – удобный и компактный элемент интерфейса, который занимает минимум места на экране и в то же время несет большую информационную нагрузку.



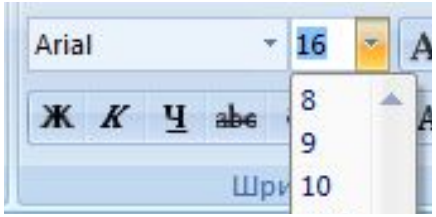




Списки: ***пролистываемыми и раскрывающимися***, причем пролистываемые могут обеспечивать как единственный (аналогично группе радиокнопок), так и множественный выбор (чекбокс); раскрывающиеся же работают исключительно как радиокнопки.

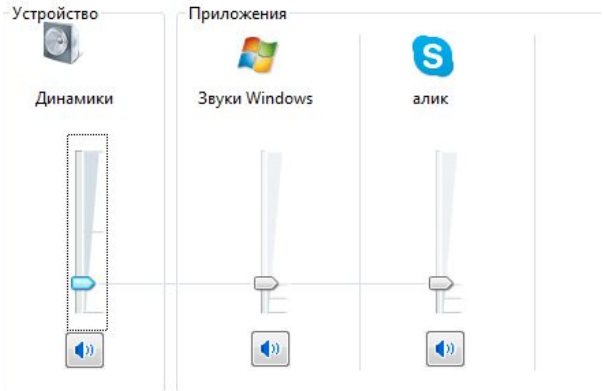
ПОЛЯ ВВОДА: КОМБОБОКСЫ, СЛАЙДЕРЫ, КРУТИЛКИ

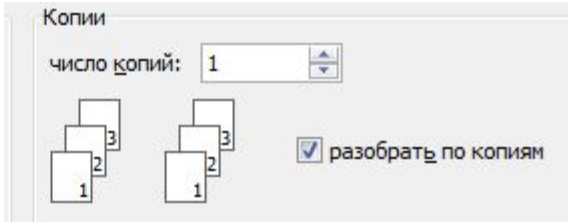
Комбобоксы. Комбобоксами (combo box), называются гибриды списка с полем ввода: пользователь может выбрать существующий элемент, либо ввести свой. Комбобоксы бывают двух видов: раскрывающиеся и расширенные.



Слайдеры (Sliders) или ползунки - элемент управления, состоящий из полосы прокрутки (slider bar), показывающей допустимо возможное значение изменяемого параметра (например, яркость или контрастность изображения), и индикатора (ползунка), показывающего текущее состояние.

Крутилка (spinner, little arrow) – элемент управления, который позволяет пользователю уменьшать или увеличивать значение некоторой величины.





ПОДХОДЫ К ПРОЕКТИРОВАНИЮ ПОЛЬЗОВАТЕЛЬСКОГО ИНТЕРФЕЙСА

Инженерно-технический подход создания графического интерфейса рассматривается на примере методики алгоритмического моделирования GOMS (от англ. «goals – operators – methods - selection rules» - «цели – действия – методы – правила выбора») (Newell, Simon, Card, 1983).

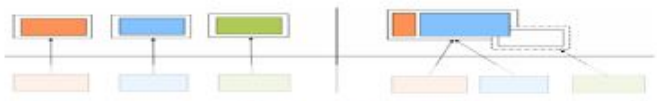
«Операторы» являются элементарными когнитивными или моторными действиями пользователя в задаче, «Методы» - процедурами достижения цели в терминах операций и подцелей, а «Правила выбора» позволяют по принципу «если-то» в каждый момент решения задачи определить направление следующего шага, исходя из текущих условий.

Ориентирован на функциональные характеристики программы, математически определяя наиболее оптимальные пути исполнения задачи.

Недостаток: невозможность анализировать сложные умственные виды деятельности пользователя, ориентируясь только на заранее определенные, внешне-наблюдаемые и последовательные действия.

Подходы к проектированию

- Инженерно-технический подход
- Когнитивный подход
- Ориентированный на пользователя (User Centered)
- Деятельностный (Activity-Centered)
- Экспертный (Genius)
- Целе-ориентированный (Goal Centered Design)



КОГНИТИВНЫЙ ПОДХОД

Когнитивный подход к проектированию ПИ, сменивший алгоритмическое моделирование, впервые стал рассматривать субъекта труда как центральную фигуру процесса взаимодействия с системой (Бродбент, 1967; Сперлинг, 1967; Линдсей, Норман, 1974; Найссер, 1981).

Учитывает не функциональные характеристики системы, как предполагалось инженерами раньше, а качество предоставления и управления информацией с точки зрения возможностей и ограничений человека (Kuutti, 1993, Kaptelinin, 1996).

Однако анализ только процессов восприятия и переработки информации оказался недостаточным для проектирования эргономичного интерфейса, поскольку не позволял определить состав и последовательность выводимой на экран информации .

ДИЗАЙН, ОРИЕНТИРОВАННЫЙ НА ПОЛЬЗОВАТЕЛЕЙ (ISO 13407)

USER CENTERED DESIGN

- Отношение пользователей к интерфейсу является главным показателем качества интерфейса.
- Работа над интерфейсом невозможна без изучения особенностей аудитории, например, уровня начальной подготовки пользователей, знаний предметной области и физиологических особенностей.

Ориентироваться на:

- пользователя текущего,
- неадаптированного,
- уже адаптированного?

ДОП не учитывает факта, что человек — очень адаптивная система.

Связан с этнографическими исследованиями. Риски: верная гарантия того, что проект с управленческой точки зрения пойдет насмарку.

ДИЗАЙН, ОРИЕНТИРОВАННЫЙ НА ЗАДАЧИ ПОЛЬЗОВАТЕЛЕЙ

TASK CENTERED DESIGN

Задача здесь — совокупность действий, в свою очередь являющихся совокупностями операций. Как правило, задачи могут быть решены несколькими разными способами, каждый из которых определяет свой набор действий.

Дело дизайнера интерфейсов — выбрать наиболее эффективное решение задачи и обеспечить её выполнение.

Включает в себя ДОП (в неявной форме): в самом деле, как мы выберем наилучшее решение для пользователей, если мы не знаем всего об этих пользователях?

Дизайн, ориентированный на задачи пользователей:

- ***может быть оценён экономически***
- ***проекты значительно более управляемы, что очень приятно менеджменту***

ДИЗАЙН, ОРИЕНТИРОВАННЫЙ НА ЦЕЛИ ПОЛЬЗОВАТЕЛЕЙ

Преодолеть лавинообразный рост числа функций помогает другая концепция — дизайн, ориентированный на мотивы пользователей

GOAL CENTERED DESIGN

Пользователи делают что-то для удовлетворения личных потребностей, иначе — мотивов; опознав эти потребности и сравнив их с задачами пользователей, мы получаем возможность лучше понять, что нужно делать.

Полезно составить список мотивов целевых пользователей и просто сравнить их со списком задач. Даже это помогает лучше оценить адекватность и полноту списка задач.

Список задач, ранжированный по частотности, приобретает собственно глубину и конкретику. Именно по этой причине в настоящее время дизайн, ориентированный на мотивы пользователей, является наиболее распространенным подходом к целеполаганию в дизайне интерфейсов (во всяком случае, обсуждается он чаще всего).

Плюс – мотивы объясняют задачи.
Минус – невозможно определить, когда кончается работа и когда начинаются самообман и шарлатанство.

ЭКСПЕРТНЫЙ ПОДХОД (GENIUS)

Команды, использующие Genius проектирование, в поиске информации для принятия решений не смотрят дальше своего собственного опыта.

Секрет в том, что вам необходим значительный опыт прошлых исследований, чтобы на нём основывать сегодняшние решения.

То, что вы раньше проектировали подобную функциональность, не считается. (Такой подход лучше было бы назвать подходом «Плавали, знаем».) Важен именно опыт проведения исследований.

Много стилей хороших и разных

Команды, которые способны обеспечить наилучшее качество проектирования хорошо знакомы со всеми стилями принятия решений и знают, как быстро переключаться между ними. Они знают, когда им необходима вся скрупулезность и точность проектирования, ориентированного на пользователей, а когда им важнее очень быстро принять решение, зная, что его последствия будут незначительными.

Арсенал методов и хорошее понимание того, как и когда их использовать

УСПЕШНЫЙ ПРОДУКТ

Постулируемые выгоды

Качественные интерфейсы

- Приводят к успеху продукта
- Снижают затраты на разработку
- Удешевляют поддержку
- Повышают конкурентоспособность
- Увеличивают лояльность клиентов
- Повышают эффект автоматизации
- Избавляют от критических ситуаций

Успех продукта

Удешевление поддержки

McAfee®
Proven Security™

количество обращений уменьшилось на **90%**

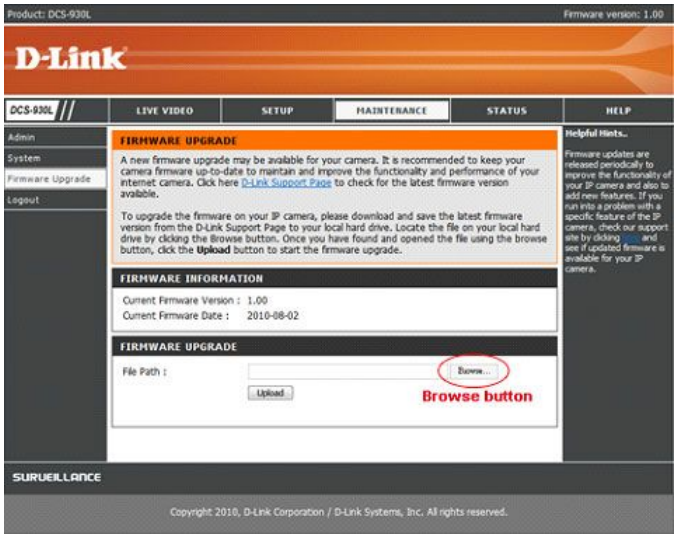
Эффективность

РОСГОССТРАХ

скорость работы увеличилась в **2,8** раза
количество ошибок снизилось **на порядок**

Лояльность

ЗНАЧИМОСТЬ ИНТЕРФЕЙСА В ЗАВИСИМОСТИ ОТ ТИПА АВТОМАТИЗИРОВАННОЙ СИСТЕМЫ



Классификация программных продуктов включает:

- Встроенное программное обеспечение
- Одноразовые программы
- Внутреннее заказное корпоративное ПО
- Массовое (коробочное) ПО
- Игры

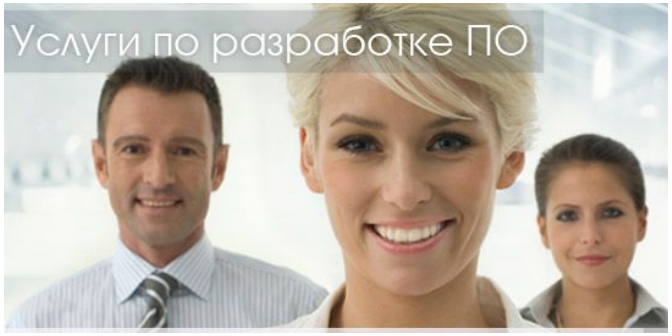
Встроенное ПО:

- Прочно связано с железом
- Редко обновляется – «второго шанса не будет»
- Требует оптимизации кода
- Средства ввода-вывода ограничены
- Нет общепризнанного стандарта элементов интерфейса

Одноразовые программы:

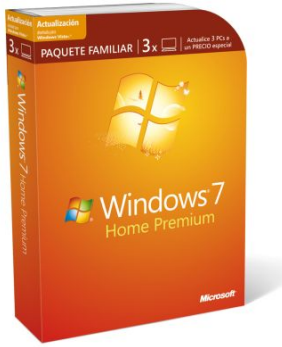
- Создается временно для решения конкретной небольш
- Обычно используется всего один раз
- Интерфейс не имеет значения

ЗНАЧИМОСТЬ ИНТЕРФЕЙСА В ЗАВИСИМОСТИ ОТ ТИПА АВТОМАТИЗИРОВАННОЙ СИСТЕМЫ



Внутреннее заказное корпоративное ПО:

- Ограниченный круг пользователей
- Высокие требования к срокам разработки
- Однородное программное окружение
- Вне зависимости от качества ПИ, пользователям придется к нему привыкнуть



Массовое (коробочное) ПО:

- Очень большое количество пользователей
- Высокая конкуренция
- Множество вариаций программного окружения
- Интерфейс должен быть максимально простым
- Много версий («в следующей версии сделаем лучше»)

Игры:

- Только одна версия! Второго шанса не будет!
- Рынок ориентирован на хиты
- Дизайн имеет решающее значение
- Массовая аудитория очень похожих пользователей
- Можно ограничивать требования к аппаратному и программному окружению

СОСТАВЛЯЮЩИЕ УСПЕШНОГО ПРОДУКТА

УСПЕШНЫЙ ПРОДУКТ:

- востребован на рынке,
- выполняет функциональное назначение,
- доставляет удовольствие,
- рекомендуют друзьям и знакомым.