**РАЗРАБОТКА ИНТЕРФЕЙСОВ ПОЛЬЗОВАТЕЛЯ**

Для зачёта:

* Сдать все лабы
* Написать все конспекты (вне зависимости болел ты или нет)
* Электронные конспекты принимаются, только если ты был на паре (лекции).

# СОДЕРЖАНИЕ

[1. ЛЕКЦИЯ ОСНОВНЫЕ ПОНЯТИЯ И ОПРЕДЕЛЕНИЯ ОБЛАСТЕЙ 5](#_Toc168126526)

[1.1 «Основные понятия предметной области» 5](#_Toc168126527)

[1.2 Пользовательский интерфейс (Uai) 6](#_Toc168126528)

[2. ЛЕКЦИЯ ВЕБ ЮЗАБИЛИТИ И ЭРГОНОМИКА 10](#_Toc168126529)

[2.1 «Веб-эргономика и юзабилити» 10](#_Toc168126530)

[2.2 «Критерии качества интерфейса пользователя» 13](#_Toc168126531)

[2.3 Количество человеческих ошибок. 15](#_Toc168126532)

[3. ЛЕКЦИЯ ПРОЕКТИРОВАНИЕ ГРАФИЧЕСКОГО ПОЛЬЗОВАТЕЛЬСКОГО ИНТЕРФЕЙСА 17](#_Toc168126533)

[3.1 Особенности графического интерфейса 17](#_Toc168126534)

[3.2 Объектный подход к проектированию интерфейса концепция интерфейса, управляемого данными 19](#_Toc168126535)

[3.3 Объекты и отношения между ними 19](#_Toc168126536)

[3.4 Компоненты графического интерфейса 22](#_Toc168126537)

[4. ЛЕКЦИЯ АФФОНИЕНС 28](#_Toc168126538)

[4.1 Введение 28](#_Toc168126539)

[4.2 Эстетика 30](#_Toc168126540)

[4.3 Время 31](#_Toc168126541)

[4.4 Психологическое напряжение 31](#_Toc168126542)

[4.5 Количественный анализ интерфейса 32](#_Toc168126543)

[5. ЛЕКЦИЯ РАЗРАБОТКА ПРОТОТИПА 35](#_Toc168126544)

[5.1 Введение 35](#_Toc168126545)

[5.2 Виды прототипов и технологии их создания 36](#_Toc168126546)

[5.3 Основные компоновочные блоки макета страницы 37](#_Toc168126547)

[6. ЛЕКЦИЯ ВИЗУАЛЬНАЯ КУЛЬТУРА ДИЗАЙНА ИНТЕРФЕЙСА 38](#_Toc168126548)

[6.1 Введение 38](#_Toc168126549)

[7. ЛЕКЦИЯ ПРОЕКТИРОВАНИЕ ОБЛИКА И ПОВЕДЕНИЯ ПОЛЬЗОВАТЕЛЬСКОГО ИНТЕРФЕЙСА 39](#_Toc168126550)

[7.1 Введение 39](#_Toc168126551)

[7.2 Ценности проектирования 39](#_Toc168126552)

[7.3 Проектирование этичного взаимодействия 40](#_Toc168126553)

[7.4 Проектирование целенаправленного взаимодействия 41](#_Toc168126554)

[7.5 Проектирование прагматичного взаимодействия 42](#_Toc168126555)

[7.6 Проектирование элегантного взаимодействия 42](#_Toc168126556)

[7.7 Шаблоны проектирования 43](#_Toc168126557)

[7.8 Типы шаблонов проектирования взаимодействия 44](#_Toc168126558)

[7.9 Определение и планирование реализации технической платформы и типа интерфейса 45](#_Toc168126559)

[7.10 Типы пользовательских интерфейсов: 45](#_Toc168126560)

[7.11 Проектирование корректного поведения 47](#_Toc168126561)

[7.12 Визуальный дизайн интерфейсов 48](#_Toc168126562)

[8. ЛЕКЦИЯ КОМПОЗИЦИЯ. ВИД И ОЩУЩЕНИЯ ПРИЛОЖЕНИЯ. 51](#_Toc168126563)

[8.1 Композиция 51](#_Toc168126564)

[8.2 Основные элементы композиции в веб-дизайне 52](#_Toc168126565)

[8.3 Гармония цвета 53](#_Toc168126566)

[8.4 Типографика 53](#_Toc168126567)

[9. ЛЕКЦИЯ УДОБСТВО ТЕСТИРОВАНИЯ 55](#_Toc168126568)

[9.1 Введение 55](#_Toc168126569)

[9.2 Имитация поведения пользователя 57](#_Toc168126570)

[9.3 Этапы юзабилити тестирования 57](#_Toc168126571)

[9.3 Особенность разработки интерфейса для автомобильных устройств 58](#_Toc168126572)

# ЛЕКЦИЯ ОСНОВНЫЕ ПОНЯТИЯ И ОПРЕДЕЛЕНИЯ ОБЛАСТЕЙ

## 1.1 «Основные понятия предметной области»

*Проектирование* – процесс создания проекта-прототипа, прообраза предполагаемого или возможного объекта, состояния, процесса.

Проектирование можно определить, как процесс разработки проекта и его фиксации в какой-либо внешне выраженной форме.

Пользовательский интерфейс представляет собой совокупность программных и аппаратных средств, обеспечивающих взаимодействие пользователя и вычислительные системы.

*Проектирование пользовательского интерфейса* – это создание тестовой версии приложения.

Это начальный этап пользовательского интерфейса (ПИ), когда распределяются функции приложения по экрану, определяются макеты экранов, содержимые элементы управления и поведения.

Пользователь при обращении с интерфейсом должен представить себе, какая информация о выполняемой задаче у него существует и в каком состоянии находятся средства, с помощью которых он будет решать данную задачу.

Эффективность работы пользователя и его интерес обеспечивает правильно сформулированная методика разработки ПИ. Именно поэтому необходимо большое внимание уделять построению ПИ (Uai) и выстраиванию пользовательского опыта в целом (Uix).

Uai – это то, как выглядит интерфейс и какие характеристики приобретает. Определяет, какого цвета будет ваше изделие, удобно ли будет человеку попадать пальцем по кнопке, читабельным ли будет текст и тому подобное.

Uai – это дизайн составляющей интерфейса. Uai-дизайнер делает графическую и текстовую информацию приятной и привлекательной, тем самым создаёт эмоциональную связь пользователя с интерфейсом.

Uix – какой опыт/впечатления получает пользователь от работы с этим интерфейсом, удаётся ли ему достичь цели и насколько просто или сложно это сделать.

Uix-проектирование ПИ – это проектирование взаимодействия пользователя с интерфейсом.

Uix-дизайнер разрабатывает сценарий того, как и каких целей может достигать пользователь, при взаимодействии с цифровым продуктом.

Uix Uai дизайн – это проектирование любых пользовательских интерфейсов, в которых удобство использования так же важно, как и внешний вид. Uix и Uai неразрывно связанны между собой и являются неотъемлемой частью успешного цифрового продукта.

## Пользовательский интерфейс (Uai)

Интерфейс пользователя представляет собой – совокупность средств и методов, при помощи которых, пользователь взаимодействует с различными устройствами и аппаратурой.

Интерфейс – только половина во взаимодействии с системой, другая половина – это пользователь.

Для хорошей работы интерфейса, нужно точно знать, что именно в любой конкретный момент, пользователь воспринимает в интерфейсе, о чём думает, чего хочет добиться.

Интерфейсы являются основой взаимодействия, всех современных информационных систем. Если интерфейс, какого-либо объекта, не изменяется, это даёт возможность, модифицировать сам объект, не перестраивая принцип его взаимодействия с другими объектами.

Пользовательские интерфейс (ПИ), делится на следующие компоненты:

1. Интерфейс командной строки – инструкции к компьютеру даются путём ввода с клавиатуры, путём тестовых команд.

В этом виде интерфейса человек «подаёт» команды компьютеру, а компьютер их выполняет и выдаёт результат человеку.

Командный интерфейс реализован в виде пакетной технологии и технологии командной строки;

1. Графический интерфейс пользователя (WIMP window image point), программные функции представляются графическими элементами экрана. Характерной особенностью этого вида интерфейса, является то, что диалог с пользователем ведётся не с помощью команд, а с помощью графических образов – меню, окон и других элементов. Отличительные особенности простого графического интерфейса – выделение областей экрана, переопределение клавиш клавиатуры, в зависимости от контекста, использование манипуляторов и клавиш клавиатуры, для управления курсором. WIMP-интерфейс характеризуется следующими особенностями:

* Вся работа с программами, файлами и документами, происходит в окнах;
* Все объекты представляются в виде значков (иконок);
* Все действия с объектами, осуществляются с помощью меню;
* Применяется широкое использование в манипуляторах, для указания на объекты.

1. Естественный языковой интерфейс (SILK интерфейс) – пользователь «разговаривает» с программой на родном ему языке. Этот вид интерфейса наиболее приближен к обычной человеческой форме общения. В рамках данного интерфейса идёт «обычный» разговор человека и компьютера, при этом компьютер находит для себя команды, анализируя человеческую речь и находя ключевые фразы. Результат выполнения команд – он также преобразует в понятную человеку форму.

Этот вид интерфейса наиболее требователен к аппаратным ресурсам компьютера, поэтому его применение началось для военных целей. Сейчас, данный вид интерфейса, широко используется, особенно для мобильных приложений. Он использует следующие технологии:

* Речевая технология. При использовании данной технологии команды подаются голосом, путём произнесения специальных зарезервированных слов-команд.

Интерфейс часто отожествляется с диалогом, который подобен диалогу или взаимодействия между двумя людьми. Речевые технологии распознают, анализируют, голос человека;

* Биометрическая технология. Здесь человек предстаёт, как совокупность признаков поведения. Картинка считывается с цифровой видеокамеры, а затем с помощью специальных программ распознавания образов, из этого изображения выделяются команды;
* Семантическая технология. Она тесно связана с искусственным интеллектом (ИИ/AI), и сходка со всеми подтипами силк (то есть с речевой технологией и биометрической).

Визуально привлекательный и удобный пользователю интерфейс – это ключевой показатель качества сайта. В сочетании с грамотной структурой и логичной навигацией по ресурсам, он привлекает посетителей и улучшает функциональность.

Главная задача в такой работе, как проектирование веб-интерфейсов – максимально упростить жизнь пользователю, сделать так, чтобы он достигал желаемых результатов, затрачивая минимум усилий.

Преимущество хорошо ПИ:

1. Повышение конкурентно способности;
2. Снижение стоимости разработки;
3. Увеличение аудитории продукта;
4. Уменьшение затрат на обучение и поддержку пользователей;
5. Уменьшение потерь продуктивности работников при внутренней системе и более быстрое восстановление утраченной продуктивности;
6. Доступность, функциональность системы для максимального количества пользователей;
7. Снижение риска ошибок (пользовательских ошибок).

Отличительные черты качественного интерфейса:

* 1. Стилевая гибкость – возможность использовать различные интерфейсы с одним и тем же приложением. На практике реализуется набором (скинс) «скинов».

Для веб-интерфейсов с помощью таблицы стилей, в том числе, возможность в выборе пользователя, собственных установок цветовой палитры;

* 1. Совместное наращивание функциональности – возможность развивать приложение без разрушения существующего интерфейса.
  2. Масштабируемость – возможность легко настраивать и расширять, как интерфейс, так и само приложение при увеличении числа пользователей, рабочих мест, объема и характеристик данных.
  3. Адаптивность к действиям пользователя. Приложение должно допускать, возможность ввода данных и команд, множеством разных способов и многовариативность доступа к прикладным функциям. Кроме этого, программа должна учитывать возможность перехода и возврат от окна к окну, от режима к режиму и правильно обрабатывать такие ситуации;
  4. Независимость ресурсов. Для создания пользовательского интерфейса должны предоставляться отдельные ресурсы, направленные на хранение и обработку данных, необходимых для поддержки пользователя.
  5. Кроссплатформенность – при переходе на другую аппаратную платформу (программную), должен осуществляться автоматический перенос и пользовательского интерфейса и конечного приложения.
  6. Мультимедийность – совокупность всех видов информации. Графической, звуковой, видео.

# ЛЕКЦИЯ ВЕБ ЮЗАБИЛИТИ И ЭРГОНОМИКА

## 2.1 «Веб-эргономика и юзабилити»

Часто термины *юзабилити* и *эргономика* употребляются в качестве синонимов. Однако, между ними есть небольшая разница: эргономичность описывает количество затраченных физических сил для работы с сайтом, а юзабилити – общую степень удобства пользования, сумму умственных усилий, требующихся от пользователя для выполнения задачи. Именно поэтому разработчики часто делают акцент на «юзабельности» сетевых проектов.

*Эргономика* – научно-прикладная дисциплина, занимающаяся изучением и созданием эффективных систем, управляемых человеком. Изучает действия человека в процессе работы, скорость освоения им новой техники, затраты его энергии, производительность и интенсивность при конкретных видах деятельности.

Современная эргономика:

1. Микро-эргономика – исследование и проектирование систем «человек–машина», в частности проектирование интерфейсов программных продуктов;

2. Миди-эргономика – изучение и проектирование систем «человек–коллектив», «коллектив–организация», «коллектив–машина» и «человек–сеть», в частности проектирование структуры организации и помещений, планирование расписания работ, гигиена и безопасность труда;

3. Макро-эргономика – исследование и проектирование системы в целом, учитывая все факторы: технические, социальные и организационные (как внешние к системе, так и внутренние). Цель – гармоничная, согласованная, надёжная работа всей системы и всех её элементов.

*Юзабилити* – степень эффективности, продуктивности и удовлетворённости, с которыми продукт может быть использован определёнными пользователями в определённом контексте использования для достижения определённых целей.

Единственное ощутимое различие между эргономикой и юзабилити – первая ставит больший акцент на технические характеристики, а вторая на процесс взаимодействия. В основе эргономики – сам продукт, в основе юзабилити – пользователь.

*Эргономичность* – количественная характеристика, описывающая количество затраченных физических сил для работы с сайтом или приложением. *Юзабилити* – качественный, описывающий сумму умственных усилий, требующихся от пользователя для выполнения задач, а также общую степень удобства пользования.

*Основная цель юзабилити* – проконтролировать, спрогнозировать и воздействовать на процессы создания системы с целью повысить конечную эргономичность продукта.

*Определённые пользователи* – интерфейс всегда создаётся для конкретной группы людей, характеризующейся своими особенностями, знаниями, навыками, ожиданиями, сильными и слабыми сторонами.

*Определённый контекст использования* – решения, которые подходят для пользователя идеально в одном контексте, могут быть совершенно неприемлемы в другом контексте.

*Определённые цели* – ни одно решение не может быть удачным, если оно не помогает достичь пользователю его целей.

Интерфейс должен быть юзабельным, но не бывает абсолютной вещи, удобной для всех. В процессе всего периода разработки интерфейса важно помнить, что именно так, как пользователь видит этот интерфейс, так он и воспринимает продукт в целом => в его понимании, если программа не удобна, то она говно бесполезна во всех своих проявлениях. Тем более это относится к сайтам и играм, т. к. конкуренция в этих нишах слишком высока, чтобы позволить себе создавать неудобные продукты.

Основная аксиома пользовательского интерфейса – хороший дизайн, который подразумевает, что программа соответствует ожиданиям пользователей о том, как она должна себя вести.

*Дизайн* – сознательные и интуитивные усилия по созданию значимого порядка. Позволяет придать определённый вид уже существующему поведению системы.

Юзабилити и дизайн важны каждому человеку, причастному к разработке, т. к. целью любого разработчика является привлечение потенциального пользователя, а пользователю важно получать наиболее позитивные чувства.

Якоб Нильсон – создатель 5 ключевых компонентов юзабилити:

1. Обучаемость – насколько быстро пользователи выполняют основные задачи и ориентируются в дизайне, сталкиваясь с ним впервые;

2. Эффективность – скорость выполнения задач и ориентация в проекте, если пользователи ранее ознакомились с дизайном;

3. Запоминаемость – простота и скорость взаимодействия пользователей с дизайном по прошествии длительного времени;

4. Ошибки – количество и качество ошибок пользователей при работе с ресурсом, а также возможность их исправления (или истребления таких пользователей);

5. Удовлетворённость – оценка эстетических качеств дизайна и степень удовольствия работы с ним.

Все эти вопросы изучает юзабилити – научно-прикладная дисциплина, способствующая повышению эффективности, продуктивности, удовлетворённости и удобства использования инструментов деятельности, в том числе программного обеспечения.

## 2.2 «Критерии качества интерфейса пользователя»

Существует 4 основных критерия качества любого интерфейса:

1. Скорость работы пользователей;
2. Скорость выполнения работы является важным критерием эффективности интерфейса.
3. Длительность состоит из следующий частей:
4. Длительность восприятия исходной информации – пользователь должен представить себе, какая информация о выполняемой у него существует и в каком состоянии находятся средства, с помощью которых он будет решать данную задачу. Основное время здесь пойдёт на считывание показаний системы;
5. Длительность интеллектуальной работы – оценивается взаимодействие пользователей с системой и состоит из 7 шагов:

* Формирование цели действий;
* Определение общей направленности действий;
* Определение конкретных действий;
* Выполнение действий;
* Восприятие нового состояния системы;
* Интерпретация состояния системы;
* Оценка результата.

Из этого списка становится видно, что процесс размышления занимает почти все время, в течение которого пользователь работает с компьютером

К сожалению, существенно повысить скорость собственного мышления невозможно, но можно уменьшить влияние факторов.

Факторы, ускоряющий процесс мышления:

1. Непосредственное манипулирование – пользователь не отдаёт команды системе, а манипулирует объектами.

2. Применение эффективных методов при потере фокуса внимания – работая с системой, пользователи постоянно отвлекаются, что требует максимальное облегчение возврата к работе и проектировать интерфейс так, чтобы пользователь меньше о нём думали. Для продолжения работы пользователь должен знать (информацию предоставляем визуально!!!):

* На каком шаге остановка?
* Какие команды и параметры уже даны системе?
* Что именно он должен сделать на текущем шаге?
* Куда было обращено его внимание в момент отвлечения?

Длительность физических действий пользователя зависит от степени автоматизации работы и степени необходимой точности работы. Любое физическое действие, совершаемое с помощью мускулатуры, может быть или точным, или быстрым. Вместе точность и быстрота встречаются исключительно редко.

Пользователь, как правило, управляет компьютером двумя способами: мышью (трекпадом) и клавиатурой. Клавиатура не требует особой точности движений – неважно, быстро нажали клавишу или медленно, равно как «сильно» или «слабо». Мышь, напротив, инерционна, именно поэтому оптимизация использования мыши в системе может повысить общую скорость работы. Мышь не предназначена для точных (1-2 пикселя) манипуляций. Поэтому лучший способ повысить доступность кнопки – сделать её большой и располагать ближе к курсору. У этого правила есть два следствия:

1. Кнопка бесконечного размера: При подведении курсора к краю экрана он останавливается, даже если движение мыши продолжается. Таким образом, кнопка, расположенная впритык к краю экрана, имеет бесконечную высоту (верхняя/нижняя граница) или ширину (левая/правая).

То есть скорость достижения такой кнопки зависит только от расстояния до неё. Кнопка в углу экрана вообще королева, т. к. имеет бесконечные размеры в обоих измерениях. Для её достижения от пользователя требуется всего лишь дёрнуть мышь в нужном направлении, не заботясь о её скорость и не предпринимая попыток остановить её в «нужном месте».

1. Нулевая дистанция до кнопки: Контекстное меню, вызываемое нажатием ПКМ всегда открывается под курсором => расстояние до любого его элемента всегда минимально. Именно поэтому контекстное меню является почти самым быстрым и эффективным элементом.

## 2.3 Количество человеческих ошибок.

Человек при работе с ПК постоянно ошибается.

Типы ошибок:

1. Ошибки недостаточного понимания предметной области;

2. Опечатки;

3. Ошибки не считывания показаний системы;

4. Моторные ошибки – ситуации, когда пользователь знает, что делать, как делать, но не может сделать из-за трудности физического действия.

Для решения ошибок есть три способа:

1. Блокировка опасных действий до подтверждения.

2. Проверка всех действий перед их принятием.

* Меню (отключение пунктов);
* Если манипулируем объектами, можно индицировать возможные действия изменением поведения объектов.

3. Самостоятельный выбор системой необходимых команд или параметров.

Средства обучения

1. Бумажная документация;

2. Оперативная справка;

3. Понятность системы: ментальная модель (надо понимать, как система работает), метафора (использование готовой модели, схожесть с другими системами), аффореданс и стандарт.

# ЛЕКЦИЯ ПРОЕКТИРОВАНИЕ ГРАФИЧЕСКОГО ПОЛЬЗОВАТЕЛЬСКОГО ИНТЕРФЕЙСА

## 3.1 Особенности графического интерфейса

В основу разработки практически любого графического пользовательского интерфейса (GUI – Graphical User Interface) положены три метафоры: «рабочий стол», «работаешь с тем, что видишь», «видишь, что получил».

Метафора «рабочий стол», в частности, означает следующее:

Для человека, сидящего за рабочим столом, доступны как определенные источники информации, так и средства обработки этой информации. При этом на рабочем столе могут находиться документы, представленные в различной форме: текст, таблицы, графики, рисунки и т. д., относящиеся либо к различным задачам, либо к разным аспектам одной и той же задачи. В любом случае человек имеет возможность просмотреть любой из этих документов, сделать пометки или выборку из них, сравнить содержащиеся в них данные.

Другие две метафоры являются развитием идеи «рабочего стола».

В каждый момент времени сидящий за столом может работать только с теми документами, которые он видит перед собой. Если необходимый документ в данный момент отсутствует на столе, его предварительно требуется достать из ящика стола, из папки или из портфеля.

В рамках графического интерфейса пользователя все три метафоры получили достаточно адекватное воплощение: пространство экрана монитора – это рабочий стол пользователя, необходимые для решения задачи объекты представлены на нем в виде соответствующих графических образов(пиктограмм и окон), а чтобы изменить рабочую среду, пользователю достаточно изменить состав объектов, представленных на рабочем столе; при этом все необходимые действия выполняются не с помощью команд, а путем прямого манипулирования объектами (точнее, их образами).

Прямое манипулирование объектами обладает следующими достоинствами:

* + Обеспечивает визуальный контроль за выполняемыми операциями.
  + Позволяет легко восстановить предшествующее состояние «рабочего стола».
  + Позволяет решать различные задачи, используя ограниченный набор стандартных операции (открытие/закрытие окна, буксировка объекта, изменение атрибутов окна или объекта и т. п.).

Еще одна важная особенность современных графических интерфейсов – это *многооконность*.

Многооконная технология обеспечивает пользователю доступ к большему объему информации, чем при использовании одного экрана. Кроме того, имея через окна доступ к нескольким источникам информации одновременно, пользователь может объединять имеющуюся в них информацию. Например, изображения, полученные с помощью графического редактора, можно включить в текстовый документ.

Таким образом, графический интерфейс расширяет пространство обзора и облегчает работу пользователя. Вместе с тем само по себе предоставление пользователю графического интерфейса еще не гарантирует повышения эффективности его работы. Это обусловлено тем, что метафора «рабочий стол» далеко не всегда означает «аккуратный рабочий стол». Если «рабочий стол» плохо организован, существует опасность, что пользователь будет тратить больше времени на работу с «папками», чем на решение стоящих перед ним задач.

## Объектный подход к проектированию интерфейса концепция интерфейса, управляемого данными

*Разработка, управляемая данными* (сокращенно DCD – Data-centered Design) означает, что проектирование интерфейса поддерживает такую модель взаимодействия пользователя с системой, при которой первичными являются обрабатываемые данные, а не требуемые для этого программные средства.

Другими словами, при таком подходе основное внимание пользователя концентрируется на тех данных, с которыми он работает, а не на поиске и загрузке необходимого приложения.

## 3.3 Объекты и отношения между ними

Рассмотренные выше особенности графических интерфейсов, а также положенная в основу их реализации DCD-технология обуславливают необходимость применения для проектирования GUI объектно-ориентированного подхода.

Такой подход предполагает использование аналогий между программными объектами и объектами реального мира. С точки зрения пользовательского интерфейса, объектами являются не только файлы или пиктограммы, но и любое устройство для хранения и обработки информации, включая ячейки, параграфы, символы, и т. д., а также документы, в которых они находятся.

Объекты, независимо от того, относятся ли они к реальному миру или имеют компьютерное воплощение, обладают определенными характеристиками, которые помогают нам понимать, что они собой представляют, и как они ведут себя в тех или иных ситуациях. Следующие понятия описывают основные аспекты и характеристики объектов, имеющих компьютерное воплощение:

* + Свойства объектов. Объекты имеют определенные характеристики или атрибуты, называемые свойствами, которые определяют их представление или возможные состояния (например, цвет, размер, дату модификации). Свойства не ограничены внешними или видимыми признаками объекта. Они могут отражать их внутреннюю организацию или текущее состояние объекта.
  + Операции над объектами. Все действия, которые могут быть выполнены с (или над) объектом, считаются допустимыми операциями. Перемещение или копирование объекта являются примерами операций. Пользователь может выполнять операции над объектами, используя те или иные механизмы, предоставляемые интерфейсом, (в частности, командное управление и прямое манипулирование).
  + Связь (отношения) между объектами. Любой объект тем или иным образом взаимодействует с другими объектами. Во многих случаях взаимоотношения между объектами могут быть описаны как связь определенного типа. Наиболее общими типами отношений являются наборы (Collection), объединения (Constraints), и композиции (Composites).

*Набор* представляет собой наиболее простой тип отношения, которое отражает наличие у объектов некоторых общих свойств. Результаты запроса (поиска) по образцу или операции множественного выбора объектов – примеры использования данного типа отношения. Важным достоинством этого типа отношения является то, что он позволяет указывать операции, которые должны относиться к определенному набору объектов.

*Объединение* отражает более «тесное» отношение между объектами, при котором изменение объекта влияет на некоторый другой объект в наборе.

Простейший пример такого отношения – изменение формата соседней страницы при добавлении текста на предыдущей странице документа.

*Композиция* имеет место в том случае, когда агрегация нескольких объектов может рассматриваться как новый объект со своим собственным множеством свойств и допустимых операций. Столбец ячеек в таблице и параграф в текстовом документе – это примеры композиций.

Еще один распространенный тип отношений между объектами – контейнер.

*Контейнер* является объектом, который содержит другие объекты (например, рисунок в документе или документ в папке могут рассматриваться как часть содержимого соответствующего контейнера). Свойства контейнера часто влияют на поведение его содержимого. Это влияние может заключаться в расширении или подавлении некоторых свойств содержащихся в нем объектов или в изменении перечня допустимых операций. Кроме того, контейнер управляет доступом к своему содержимому, а также преобразованием типа (формата) включаемого в него объекта. Это, в частности, может сказаться на результате пересылки объекта из одного контейнера в другой.

Первым шагом в объектно-ориентированном проектировании интерфейса должен быть анализ целей пользователей и особенностей выполняемых ими заданий. При проведении такого анализа следует определить основные компоненты или объекты, с которыми взаимодействует пользователь, а также характерные особенности объектов каждого типа. Необходимо также выявить перечень операций, выполняемых над объектами, их влияние на состояние и свойства объектов.

После завершения анализа можно переходить к описанию возможных способов взаимодействия пользователя с объектами различных типов. На этом шаге выбирается форма визуального представления объектов. При этом следует иметь в виду, что визуальный образ объекта в зависимости от ситуации может изменяться. Например, контейнер может быть представлен и в виде пиктограммы, и в виде окна, отображающего содержимое этого контейнера.

Следующим этапом проектирования GUI является компоновка и пространственное размещение на экране визуальных элементов интерфейса. Именно на этом этапе должны быть решены такие проблемы, как выбор цвета, размера и других атрибутов этих элементов, а также выбор средств и методов привлечения внимания пользователя к наиболее важной информации, отображаемой на экране.

## 3.4 Компоненты графического интерфейса

*Рабочий стол* предоставляет пользователю первичную рабочую область; он заполняет экран и формирует визуальный фон для всех выполняемых операций. Тем не менее, Рабочий стол является не просто фоном. Он может также быть использован в качестве основы для размещения объектов файловой системы.

Кроме того, для компьютера, подключенного к сети, Рабочий стол служит в качестве частной рабочей области, через которую пользователь может получить доступ к другим объектам сети.

Пиктограммы используются для визуального представления на экране объектов или задач. Как правило, это небольшие законченные рисунки, отображающие сущность представляемых объектов или явлений. Поскольку пиктограммы являются одним из основных средств взаимодействия пользователя с приложением, важно не только обеспечить поддержку существующих (системных) пиктограмм, но и разработать новые; от того, насколько они будут соответствовать своему предназначению, будет зависеть и эффективность работы пользователя.

В реальном мире взгляд через разные окна позволяет получить различные изображения внешнего мира. Аналогичную роль играют окна и в графическом интерфейсе.

Другими словами, окно является средством просмотра и редактирования информации, а также отображения содержимого и свойств объектов. Окна могут использоваться также для вывода на экран значений параметров, результатов выполнения команд, наборов инструментов и сообщений, информирующих пользователя о конкретной ситуации.

Пользователь может взаимодействовать с объектами приложения, используя различные устройства ввода. Наиболее распространенные из них – мышь, клавиатура и перо.

Мышь является основным устройством ввода при использовании графического интерфейса. Другие типы устройств указания, которые эмулируют мышь (например, трэкболлы), также подпадают под этот общий термин.

Мышь функционально связана с графическим символом на экране, который называется *указателем*. Позиционируя указатель и нажимая или щелкая кнопку мыши, пользователь может выбирать объекты и операции.

По мере того, как пользователь перемешает указатель через экран, его форма может изменяться, чтобы обеспечить информирование пользователя (*обратную связь*) о конкретной позиции, операции или состоянии.

Каждый указатель имеет определенную точку, называемую *горячей точкой*, которая идентифицирует точную позицию указателя мыши на экране. Горячая точка определяет, на какой объект воздействует пользователь посредством мыши. Для объектов на экране может быть также определена горячая зона; *горячая зона* – это область, в пределах которой должна (или может) находиться горячая точка, чтобы считаться расположенной над объектом. Как правило, горячая зона совпадает с границами объекта, но может быть как больше, так и меньше, чтобы облегчить работу пользователя.

Взаимодействие пользователя с приложением посредством мыши основано на использовании левой и правой кнопок мыши (в дальнейшем соответственно ЛКМ и ПКМ). Действия, выполняемые с помощью ПКМ, обычно дублируют функции, которые доступны через ЛКМ, но могут быть реализованы альтернативным способом. Система позволяет пользователю изменять распределение кнопок.

Для большинства операций, выполняемых с помощью мыши, нажатие кнопки только идентифицирует начало выполнения операции. Сама же операция выполняется при освобождении кнопки. Исключением является функция *автоповторения*. Например, установив указатель на стрелку полосы прокрутки и удерживая нажатой кнопку мыши, можно обеспечить непрерывную прокрутку до тех пор, пока кнопка не будет отпущена.

Теоретически, возможны и. другие приемы работы с мышью, например, *аккорд* (нажатие нескольких кнопок мыши одновременно) и *мультищелчок* (тройной или даже «учетверенный» щелчок), однако они требуют наличия соответствующих моторных навыков пользователя и на практике используются очень редко.

Клавиатура – это основное средство ввода или редактирования текстовой информации. Тем не менее, при реализации графического интерфейса она может использоваться для ввода других типов данных, а также для управления, переключения режимов и как средство ускоренного доступа к объектам и операциям.

С точки зрения организации взаимодействия пользователя с объектами приложения все клавиши можно разделить на следующие функциональные группы:

* + текстовые клавиши и клавиши пунктуации;
  + клавиши навигации;
  + функциональные клавиши;
  + клавиши доступа;
  + клавиши режима;
  + клавиши-акселераторы.

*Клавиша доступа* (иногда именуемая также мнемонической клавишей) – это текстовая клавиша, которая при использовании в комбинации с клавишей <Alt> активизирует соответствующий элемент управления. Клавиша доступа должна соответствовать одному из символов текстовой метки этого элемента управления. Например, комбинация клавиш <Alt> + О может активизировать элемент управления, имеющий метку Открыть. Эффект активизации элемента управления зависит от типа этого элемента.

Клавиша доступа должна быть уникальной в пределах своей области действия (т. е. одна и та же клавиша не должна использоваться для доступа более чем к одному элементу управления, расположенному в этой области). В противном случае с помощью данной клавиши всегда будет активизироваться только один элемент – расположенный первым в этой области.

*Клавиши режима* изменяют способ действия других клавиш (или других устройств ввода). Различают два типа таких клавиш: клавиши-переключатели и клавиши-модификаторы.

*Клавиша-переключатель* включает или выключает конкретный режим при каждом очередном нажатии. Например, нажатие клавиши <Caps lock> приводит к переключению с верхнего регистра на нижний и наоборот.

*Клавиша-модификатор*, в отличие от клавиш-переключателей, устанавливает режим, который остаётся в силе, только пока клавиша-модификатор удерживается в нажатом состоянии. К ним относятся клавиши <Shift>, <Ctrl> и <Alt>. Их использование, во многих случаях, оказывается предпочтительнее из-за того, что они требуют привлечения внимания пользователя к выбору режима и, вместе с тем, позволяют легко отменить выбранный режим.

*Клавиши-акселераторы* (называемые также горячими клавишами) – это клавиши или комбинации клавиш, которые обеспечивают быстрый доступ к часто выполняемым операциям. В качестве таких комбинаций рекомендуется использовать <Сtrl> +<символ> и функциональные клавиши (с Fl no F12). По определению, клавиши-акселераторы являются «клавиатурным эквивалентом» других элементов пользовательского интерфейса. Исходя из этого, избегайте использования клавиши-акселератора как единственного средства доступа к какому-либо объекту или функции.

При назначении клавиш-акселераторов следует учитывать приведенные ниже рекомендации:

* + Используйте комбинацию <Shift>+<клавиша> для расширения или дополнения действия, выполняемого с помощью этой <клавиши> без <Shift>. Например, если комбинация клавиш <Alt>+<Tab> обеспечивает переключение между окнами сверху вниз, то комбинация <Shift>+<Alt>+<Tab> переключает окна в обратном порядке.
  + Используйте комбинации <Сtrl> +<клавиша> для усиления эффекта действия, выполняемого с помощью <клавиши>. Например, при редактировании текста клавиша <Ноmе> обеспечивает переход на начало строки, а комбинация <Ctrl>+<Home> – в начало текста.
  + Избегайте комбинаций <Alt>+<клавиша>, поскольку они могут конфликтовать со стандартным клавиатурным доступом к меню и элементам управления интерфейса. Комбинации <Alt>+<Tab>, <Alt>+<Esc> и <Alt>+<Spacebar> зарезервированы для системного использования; комбинации <Аlt> + <цифра> обеспечивают ввод специальных символов.
  + Учитывайте назначения клавиш-акселераторов, использованные разработчиками системного программного обеспечения. Например, в ОС MS Windows комбинация <Ctrl>+C используется для выполнения команды Copy (Копировать).
  + Предоставляйте пользователю право изменять назначения клавиш-акселераторов в вашем приложении, когда это возможно.
  + Используйте клавишу <Esc> для прерывания выполняемой операции; как правило, она также используется в качестве клавиши-акселератора для выполнения команды Cancel (Отменить).
  + Сделайте ваше приложение нечувствительным к случайным (ошибочным) комбинациям клавиш.

Некоторые клавиатуры содержат три новых клавиши: клавишу <Application> (Приложение) и две клавиши <Windows> (Окна). Основное назначение клавиши <Application>– вызов всплывающего меню для текущего выбора (т. е. она аналогична комбинации <Shift>+F10). Вы можете также использовать ее совместно с клавишами-модификаторами для выполнения специализированных функций приложения. Нажатие любой из клавиш <Windows> – левой или правой – приводит к появлению меню. Эти клавиши также используются системой как модификаторы для специфических системных функций. Не используйте эти клавиши в качестве модификаторов для функций несистемного уровня.

# ЛЕКЦИЯ АФФОНИЕНС

## 4.1 Введение

В современном значении этого термина аффорденсом называется ситуация, при которой объект показывает субъекту способ своего использования своими неотъемлемыми свойствами.

Польза Аффорденса заключается в том, что он позволяет пользователям обходиться без предварительного обучения, благодаря чему аффорденс является самым эффективным и надёжным средством обеспечения понятности.

Распространённые примеры:

1. Стрелки выпадающего меню;
2. Выпуклая кнопка;
3. Файлы и папки (включая корзину) на рабочем столе.

Виды аффорденса:

1. Явные или прямые указатели – призваны показать, как посетитель сайта или пользователь приложения может совершить нужное ему действие;

Способы применения:

* Предложение купить;
* Оформить заказ или подписку;
* Посмотреть товары или перейти в корзину;

Чтобы не перегрузить страницы, можно использовать сочетание текста аффорденса с визуальным элементом.

1. Метафорические или символические указатели – предполагается, что пользователь поймёт назначение элементов, потому что видел его раньше на других сайтах или в других приложениях;

Примеры:

* Изображение конверта в поле подписки на рассылку;
* Виджет с телефонной трубкой;

Чаще всего такие аффорденсы используются для оформления главного меню и ссылок на типовые разделы сайта, поэтому важно правильно подобрать изображения. При этом символические указатели могут одинаково восприниматься независимо от тематики и контента;

1. Скрытые указатели – такие подсказки или элементы навигации появляются после совершения какого-либо действия, чаще всего – наведения курсора. Они призваны спрятать дополнительные функции, которые перегрузили бы страницу или отвлекли бы внимание пользователя от главного контента. Использовать такие аффорденсы следует осторожно: если действия, вызывающие появление указателя, совершается редко – пользователь так и не узнает о скрытой возможности;
2. Негативные указатели – назначение этих подсказок – показать, что пункт меню или иной элемент в настоящее время неактивен.

Такое оформление достаточно распространено и привычно для большинства пользователей, поэтому не рекомендуется выбирать подобное оформление для кнопок и полей, которые всё-таки активны и доступны для взаимодействия;

На одном сайте могут присутствовать несколько видов аффорденсов, и каждый из них будет полезен посетителям, если действительно находится на своём месте. При использовании аффорденсов важно не допускать двоякого толкования и учитывать опыт аудитории. Контекст аффорденса позволяет использовать символические аффорденсы нетипичным образом при условии, что посетители сайта формируются в правильный ассоциативный ряд.

Способы передачи аффорденсов:

1. Построение конфигурации объектов конфигурацией элементов управления – работает хорошо в реальном мире, но не на экране;
2. Видимая принадлежность управляющих элементов объекту;
3. Визуальное совпадение аффорденса экранных объектов с такими же аффорденсами объектов реального мира;
4. Изменение свойств объекта при подведении к нему курсора; Способ обучения – стандарт.

Если что-либо нельзя сделать «самопроизвольно» понятным, всегда можно сделать это везде одинаковым, чтобы пользователи обучались единожды.

Дизайн элементов управления, выдержанный в едином ключе для различных программ, помогает пользователю обучиться работать с новой программой.

Популярность может быть достигнута одним из способов:

* Он, может быть, во всех системах;
* Он может быть популярен внутри отдельной системы.

Субъективное удовлетворение пользователя:

1. Субъективные ощущения эстетики;
2. Субъективные ощущения времени (СОВ);
3. Субъективные ощущения псих напряжения (СОПН);
4. Субъективные ощущения собственной глупости;
5. Субъективные ощущения самовыражения.

## 4.2 Эстетика

Программисты заботятся о функциональности приложения, однако его внешний вид его может влиять на подсознательное поведение пользователя. Если приложение решает серьёзные задачи, выглядит навязчивым и легкомысленным, то пользователи могу подвергнуть сомнению его надёжность. Чтобы интерфейс был эстетически привлекательным, он должен быть незаметным в процессе его использования. Для этого:

1. Избегаем развязности в изображении. Лучше скромный интерфейс;
2. Избегаем ярких цветов;
3. избегаем острых углов в изображениях;
4. Стараемся сделать изображение «лёгким» и «воздушным».

## 4.3 Время

Любой человек хочет работать быстро. Если работу можно выполнить быстро, у человека возникают приятные ощущения. Однако, СОВ зачастую сильно отличается от объективного, так что методы повышения реальной скорости работы помогают не всегда. Воспринимаемая продолжительность действий напрямую зависит от уровня активности пользователя. Субъективную скорость работы можно повысить двумя способами:

1. Заполнение пауз между событиями – Показывать индикатор степени выполнения – чем больше информации, тем меньше субъективное время. С другой стороны, эта информация может вызвать стресс кратковременной памяти;
2. Разделение крупных действий пользователя на более мелкие, при этом количество работы увеличивается, но субъективная длительность снижается. Увеличивает усталость.

## 4.4 Психологическое напряжение

Пользователь знает, что во время работы может что-то испортить, и это вызывает у него психологическое напряжение. Чтобы не вводить пользователя в стресс, необходимо иметь возможность:

1. Отменять свои предыдущие действия без ограничения количества уровней отмены и типа отменяемых действий;
2. Прятать от пользователя опасные для него места интерфейса;
3. Обеспечить чувство власти над компьютером и ПО.

Ни один пользователь не может полноценно работать с системой, которая «говорит, что пользователь глуп», то есть показывает ошибки, связанные с конкретными действиями пользователя.

Сообщения об ошибках:

1. Избегаем специализированных терминов, пишем сообщения общепонятным языком.
2. Сразу обращать внимание пользователя на конкретные поля ввода (где возникли ошибки)
3. Идеальное сообщение об ошибке отвечает на три вопроса:

* В чём проблема?
* Как исправить проблему сейчас?
* Как не допускать проблему в будущем?

## 4.5 Количественный анализ интерфейса

Существует множество методов количественного анализа. Наиболее известные приведены ниже:

1. Модель GOMS;
2. Разработка;
3. Критерий эффективности Раскина;
4. Закон Хика – время реакции при выборе из некоторого числа альтернативных сигналов зависит от их числа;
5. Закон Фитса – время, требуемое для позиционирования на какой-либо элемент, есть функция от расстояния до этого элемента и его размеров. Время достижения цели прямо пропорциональна дистанции до цели и обратно пропорциональна размеру цели. Обычно сопровождается законом Хика.

При проектировании интерфейсов закон Хика помогает определить оптимальное число элементов в однородном массиве (например, в меню). Выбор из одного меню (8 элементов) производится быстрее, чем из двух меню по 4 элемента. Общая рекомендация – $7 \pm 2$. Законы Хика и Фитса описывают действия, которые обычно следуют одно за другим, сначала определяемся с выбором (Хика), а затем попасть на нужный элемент (Фитса). Таким образом общее время можно вычислить как сумму значений двух формул. В контексте UX это значит следующее: одно длинное меню или расположение однородных элементов в одном блоке удобнее для пользователя, чем два или несколько отдельных. При проектировании интерфейса нужно учитывать оба закона и стараться оптимизировать как размеры и положение блоков, так и количество элементов в каждом блоке.

Особенно заметен закон Хика при заполнении полей форм. Один из лучших подходов к количественному анализу моделей интерфейсов – классическая модель GOMS (Goals, Objects, Methods, Selection rules). Такое моделирование позволяет предсказать, сколько времени потребуется опытному пользователю на выполнение конкретной операции при использовании данной модели интерфейса. Эта модель основана на оценке скорости печати. Время на выполнение какой-то задачи системы «Пользователь – Компьютер» является суммой всех временных интервалов, которые потребовались системе на выполнение элементарных жестов, составляющих данную задачу.

Информационная производительность интерфейса E определяется как отношение минимального количества информации, необходимого для выполнения задачи, к количеству информации, которую должен ввести пользователь. Этот параметр может изменяться в пределах от 0 до 1. Если для выполнения задачи не требуется никакой работы или работа просто не производится, то это отношение равно единице. E = 0 в случаях, когда пользователь должен ввести информацию, которая бесполезна для выполнения задачи.

# ЛЕКЦИЯ РАЗРАБОТКА ПРОТОТИПА

## 5.1 Введение

Результатом выполнения этапа предварительного проектирования является полная функциональная схема, описывающая все взаимодействие пользователя с системой. Известно, сколько экранов(страниц) нужно, и что должно находиться на каждом экране.

*Разработка прототипа* – это средство , позволяющее проанализировать идеи, прежде чем потратить на них время и деньги.

Самая веская причина – экономия времени и ресурсов.

По сравнению с реальным продуктом, прототипы простые и недорогие в разработке.

*Прототип страниц* – схематичное представление всех элементов страницы и их взаимного расположения. Прототипы используются для достижения одной или нескольких из следующих целей:

1. Проработать дизайн

2. Построить общую коммуникационную платформу

3. Увлечь других людей вашими идеями

4. Проверить техническую реализуемость

5. Протестировать проектировочные идеи с помощью пользователя.

В большинстве случаев прототип после тестирования, оказывается неправильным, и его приходится переделывать. Поэтому не следует перебарщивать с его проработкой

В прототипировании выделяют 4 стадии:

1. Концептуальный дизайн – для исследования различных метафор в интерфейсе и подходов к дизайну

2. Дизайн взаимодействия - для организации структуры экранов или страниц и переходов между ними

3. Дизайн экранов – разработка внешнего вида каждого из конкретных экранов или страниц

4. Тестирование – оценка прототипов для их последующего итерационного улучшения

## 5.2 Виды прототипов и технологии их создания

Статичные прототипы: включают в себя такую популярную методику, как бумажное прототипирование (создание скелетов) – создание прототипов при помощи графических редакторов в виде статичных изображений

Использование прототипирования на бумаге заключается в исключительной простоте модификации по результатам тестирования.

Прототип должен создаваться быстро. Его основная функция – показать часть функциональности в простейшем виде без акцентирования внимания на иконках, цветах , шрифтах

Создание схемы для каждой страницы перехода – это является промежуточным этапом между бумажным и электронным вариантом прототипа.

Динамические прототипы: Дальнейшим развитием статической раскадровки является динамическая раскадровка с применением средств анимации и т. д.

На этом этапе создается действующая модель пользовательского интерфейса. Существуют следующие подходы к прототипированию: ЧБ прототип на основе схем страниц и цветной прототип, основанный на принятом заказчиком визуальном дизайне системы.

## 5.3 Основные компоновочные блоки макета страницы

К основным блокам можно отнести:

* навигационный;
* информационный;
* сервисные;
* дизайнерские;
* рекламные.

Навигационный блок: дизайн навигации должен одновременно решать три задачи:

1. Предоставлять пользователям способ попасть из одной точки страницы в другую.

2. Отображать взаимоотношения между внутренними элементами навигации

3. Отражать связь между содержательной стороной навигации и страницей.

*Навигационные блоки*:

* на главную страницу;
* поиск и быстрый переход;
* горизонтальное меню;
* вертикальное меню;
* вторичная навигация;
* навигация по выборке;
* авторизация;
* подвал.

*Навигационная строка*. Информационный блок:

* содержание;
* текущая информация;
* раздел;
* изображения (галерея).

Сервисный блок :

* Выбор языка
* Пустой блок
* Версия для печати

Дизайнерский блок:

Рекламный блок

1. Название и слоган

2. Копирайт

# ЛЕКЦИЯ ВИЗУАЛЬНАЯ КУЛЬТУРА ДИЗАЙНА ИНТЕРФЕЙСА

## 6.1 Введение

*Дизайн* – это сознательные интуитивные усилия по созданию значимого порядка.

При проектировании интерфейса необходимо учитывать внешний вид( форму) и информационное наполнение (содержание/контент) разрабатываемого интерфейса.

Поэтому необходимо сочетать подходы различного проектирования:

1. Проектирование взаимодействия, основное внимание на поведение.

2. Информационная архитектура (занимается структурированием содержания)

3. Графический дизайн, отвечает за внешний вид интерфейса и его функционал

Визуальный дизайн интерфейсов выполняет две задачи :

1. Задачи графического дизайна

2. Задачи визуального информационного дизайна

# ЛЕКЦИЯ ПРОЕКТИРОВАНИЕ ОБЛИКА И ПОВЕДЕНИЯ ПОЛЬЗОВАТЕЛЬСКОГО ИНТЕРФЕЙСА

## 7.1 Введение

*Принципы проектирования взаимодействия* – это рекомендации, касающиеся поведения, формы и содержания продукта. Они поддерживают проектирование такого поведения продуктов, которое служит потребностям и целям пользователей. Эти принципы представляют собой набор правил, которые основаны на ценностях проектировщиков, и на их опыте, связанном с воплощением этих ценностей в жизнь. В основе таких ценностей – мысль о том, что технология должна служить человеку (а не наоборот) и что опыт общения человека с технологией должен складываться согласно возможностям человеческого восприятия и познания.

Принципы применяются на всем протяжении процесса проектирования. Они помогают преобразовывать задачи и требования, возникающие в ходе разработки сценариев, в поведенческие реакции интерфейса.

## 7.2 Ценности проектирования

Принципы – это правила, ведущие к действиям и обычно опирающиеся на ряд ценностей и убеждений. Приведенный ниже набор ценностей создали Роберт Рейман, Хью Дабберли (Hugh Dubberly), Ким Гудвин, Дэвид Фор (David Fore) и Джонатан Корман (Jonathan Korman). Он применим к любой дисциплине проектирования, которая служит потребностям человека.

Задача проектировщиков взаимодействия – создавать такие проектные решения, которые:

* Этичны [тактичны, заботливы]: не причиняют вреда и улучшают положение человека.
* Прагматичны [жизнеспособны, осуществимы]: помогают организации, внедряющей ваши проектные решения, достигать своих целей; учитывают требования бизнеса и технические требования.
* Целенаправленны [полезны, применимы]: помогают пользователям решать их задачи и достигать целей, учитывают контексты и возможности пользователей.
* Элегантны [эффективны, искусны, вызывают эмоции]: представляют собой простые, но полноценные решения, обладают внутренней (самоочевидной, понятной) целостностью, учитывают и пробуждают эмоции и познавательные процессы.

## 7.3 Проектирование этичного взаимодействия

С этическими вопросами проектировщики взаимодействия сталкиваются, когда их просят спроектировать систему, оказывающую серьезное влияние на жизнь людей. Это может быть непосредственное воздействие на пользователей системы или косвенное влияние на людей, жизнь которых каким-либо образом оказывается затронута системой.

Для проектировщиков взаимодействия этот вопрос может быть особенно животрепещущим, поскольку результаты их работы не просто убедительная передача определенной стратегии продвижения продукта. От их работы в действительности зависит реализация этой стратегии, или собственно создание продукта. Интерактивные продукты обладают возможностью изменять мир, и проектировщики должны стремиться к тому, чтобы это были изменения к лучшему. Относительно легко спроектировать систему, которая хорошо относится к пользователям, однако вычислить, какое косвенное влияние система оказывает на других людей, гораздо сложнее.

*Не навреди* – один из важнейших этических принципов проектирования взаимодействия. Продукты не должны никому причинять вреда – или, принимая во внимание сложность реального мира, должны минимизировать ущерб.

Чтобы результатом проектирования стало действительно этичное взаимодействие, мало просто не вредить – требуется способствовать. Приведем примеры улучшений, которым в широком смысле могут содействовать интерактивные системы:

* улучшение понимания (индивидуального, социального, культурного);
* повышение эффективности/действенности отдельных личностей и групп;
* совершенствование коммуникаций – как между отдельными личностями, так и между группами людей;
* снижение социокультурной напряженности между личностями и группами;
* умножение справедливости (финансовой, социальной, правовой);
* сглаживание культурных противоречий путем стимулирования общественной толерантности.

## 7.4 Проектирование целенаправленного взаимодействия

*Целенаправленность* – это не только понимание целей пользователей, но и осознание их ограничений. В этом смысле персонажи служат качественной меркой, поскольку шаблоны поведения, которые вы сможете наблюдать в ходе исследований и при создании персонажей, дадут вам хорошее представление о сильных и слабых сторонах пользователей. Целеориентированное проектирование помогает проектировщикам создавать продукты, которые поддерживают пользователей в том, в чем они слабы, и делают их более производительными в том, в чем они сильны.

## 7.5 Проектирование прагматичного взаимодействия

Чтобы иметь ценность, проектирование должно воплощаться в продукте. Будучи созданным, продукт должен увидеть свет. А увидев свет, он должен приносить прибыль создателям.

Крайне важно, чтобы цели бизнеса и технические требования учитывались в ходе проектирования. Это не означает, что проектировщики должны принимать как должное все, что говорят заинтересованные лица и разработчики. Необходим живой диалог, в котором участвуют представители бизнеса, инженерная часть команды, проектировщики, позволяющий договориться о том, какие области определения продукта заданы жестко, а какие являются гибкими. Проектирование оказывается наиболее эффективным, когда существует взаимное доверие и уважение между проектировщиками, бизнесменами и инженерами.

## 7.6 Проектирование элегантного взаимодействия

Элегантность в словаре определяется как «грациозность и сдержанная красота стиля» или «научная точность, аккуратность и простота». Элегантность в проектировании – или, по меньшей мере, в проектировании взаимодействия – включает оба этих идеала.

Создавай простые, но полноценные решения.

Один из классических элементов качественного дизайна – это экономия формы, достижение большего меньшими затратами. Это касается и поведения: следует дать пользователю простой набор инструментов, позволяющий ему добиваться великолепных результатов. Такой подход хорошо знаком программистам, которые признают, что лучшие алгоритмы коротки и понятны.

Добивайся внутренней целостности.

Качественно спроектированный продукт оставляет ощущение единого целого, в котором все составные части сбалансированы и гармоничны.

Некачественно спроектированный продукт или продукт, вообще не проектировавшийся, обычно создает впечатление, будто его наспех сколотили из выбранных случайным образом разрозненных кусков.

Учитывай и пробуждай эмоции и познавательные процессы.

Многие проектировщики традиционной школы часто говорят о желаниях и важности желаний при проектировании коммуникаций и продуктов. Однако делая такой сильный упор на одну эмоцию, они временами теряют из виду полную картину. Желание – ограниченная эмоция, если речь идет о продукте, у которого есть определенное назначение, особенно узкопрофессиональное. Пользователь должен получать стимулы и поддержку при работе с продуктом, поэтому вместо желания лучше использовать элегантность.

## 7.7 Шаблоны проектирования

Шаблоны проектирования решают целые классы проблем проектирования, возникающие путем выявления и обобщения ценных проектных находок. Деятельность по формализации знания и фиксации наилучших решений в области проектирования служит многим важным целям:

* сократить время и усилия, затрачиваемые на проектирование в новых проектах;
* повысить качество проектных решений;
* способствовать улучшению коммуникации между проектировщиками и программистами;
* повысить профессиональный уровень проектировщиков.

Шаблоны всегда применяются в рамках некоторого контекста и конструируются так, чтобы быть применимыми в типичных ситуациях, которые имеют схожий контекст использования, схожие ограничения и условия. Описывая шаблон, важно четко задать ситуацию, в которой применимо решение, дать один или несколько конкретных примеров, перечислить абстрактные признаки, характерные для всех примеров, а также рассуждения, объясняющие, почему решение является хорошим. Шаблоны проектирования не являются рецептами или готовыми решениями.

## 7.8 Типы шаблонов проектирования взаимодействия

Шаблоны проектирования взаимодействия можно выстроить в иерархию. Их можно применять на различных уровнях инфраструктуры интерфейса:

1. Шаблоны позиционирования могут применяться на концептуальном уровне и помогают определить тип продукта в отношении к пользователю.

2. Структурные шаблоны решают проблемы, связанные с управлением отображением информации и функциональных элементов на экране.

3. Поведенческие шаблоны решают широкий спектр проблем, относящихся к конкретным взаимодействиям с теми или иными элементами интерфейса.

## 7.9 Определение и планирование реализации технической платформы и типа интерфейса

Техническая платформа определяет тип оборудования, на котором можно установить информационную технологию. Она имеет сложную структуру. Главным компонентом технической платформы является тип компьютера, определяемый типом процессора: Macintosh, Atary, Sincler, Intel, J2EE т. д. Многие современные информационные технологии используют добавочное оборудование. Например, сетевые информационные технологии зависят от сетевого оборудования: модемов, адаптеров, каналов связи и т. д. В технологии мультимедиа используются приводы CD-ROM, видеокарты, звуковые карты. А так как технология мультимедиа может быть использована в сетях ЭВМ, она также зависит и от сетевого оборудования. Поэтому добавочное оборудование также входит в состав технической платформы.

## 7.10 Типы пользовательских интерфейсов:

1) Командный интерфейс. Он называется так потому, что в этом виде интерфейса человек подает «команды» компьютеру, а компьютер их выполняет и выдает результат человеку. Командный интерфейс реализован в виде пакетной технологии и технологии командной строки.

*Пакетная технология*

Вначале накапливаются данные, и формируется пакет данных, а затем пакет последовательно обрабатывается рядом программ. Недостатки этого режима – низкая оперативность принятия решений и обособленность пользователя от системы.

*Технология командной строки*

При этой технологии в качестве способа ввода информации обычно служит клавиатура, а дисплей средством вывода. Команды набираются в командной строке.

2) WIMP – интерфейс (Window – окно, Image - образ, Menu – меню, Pointer – указатель). Хотя и в этом интерфейсе машине подаются команды, но это делается «опосредственно», через графические образы. Этот вид интерфейса реализован на двух уровнях технологий: простой графический интерфейс и «чистый» WIMP интерфейс.

Простой графический интерфейс. Отличительные особенности этого интерфейса:

* Выделение областей экрана.
* Переопределение клавиш клавиатуры в зависимости от контекста.
* Использование манипуляторов и серых клавиш клавиатуры для управления курсором.

*WIMP*

Этот подтип интерфейса характеризуется следующими особенностями:

* Вся работа с программами, файлами и документами происходит в окнах;
* Все программы, файлы, документы, устройства и другие объекты представляются в виде значков;
* Все действия с объектами осуществляются с помощью меню;
* Широкое использование манипуляторов для указания на объекты.

3) SILK – интерфейс (Speech – речь, Image – образ, Language – язык, Knowlege – знание). Компьютер находит для себя команды, анализируя человеческое поведение.

*Речевая технология*

При этой технологии команды подаются голосом путем произнесения специальных зарезервированных слов-команд.

*Биометрическая технология*

Здесь человек предстаёт как совокупность признаков поведения. Картинка считывается с цифровой видеокамеры, а затем с помощью специальных программ распознавания образов из этого изображения выделяются команды.

*Семантический интерфейс*

Об этой технологии известно крайне мало. Похоже, что она тесно связана с искусственным интеллектом и сходна со всеми подтипами SILK и другими типами тоже. Возможно, что в связи с важным военным значением этих разработок эти направления были засекречены.

## 7.11 Проектирование корректного поведения

Главный вопрос для маркетологов и проектировщиков заключается в том, как влиять на поведение людей, и какие подходы к проекту способствуют таким изменениям.

Стивен Вендель (Stephen Wendel), главный научный сотрудник HelloWallet, в 2014 году написал книгу Designing for Behavior Change (Проектирование, изменяющее поведение). Это исследование путей применения психологии и поведенческой экономики для проектирования продуктов (product design).

В своей книге Вендель вводит 4 стадии проектирования для коррекции поведения пользователей: Понимание, Ознакомление, Проектирование, Улучшение.

* + Понимание – изучение того, как люди принимают решения и как наши когнитивные механизмы способствуют (или препятствуют) изменению поведения.
  + Ознакомление – компания формулирует, чего она хочет достичь со своим продуктом и для кого это делается.
  + Проектирование – разработку дизайна можно разбить на две подзадачи: (1) проектирование общей концепции продукта; (2) разработка пользовательского интерфейса.
  + Улучшение – анализируйте полученные данные, дабы формулировать идеи для постоянного усовершенствования продукта.

## 7.12 Визуальный дизайн интерфейсов

Визуальный дизайн интерфейсов – очень нужная и уникальная дисциплина, которую следует применять в сочетании с проектированием взаимодействия и промышленным дизайном. Она способна серьезно повлиять на эффективность и привлекательность продукта, но для полной реализации этого потенциала нужно не откладывать визуальный дизайн на потом, а сделать его одним из основных инструментов удовлетворения потребностей пользователей и бизнеса.

Дизайнеры создают объекты, которыми будут пользоваться другие люди. Если говорить о дизайнерах визуальных интерфейсов, то они ищут наилучшее представление, доносящее информацию о поведении программы, в проектировании которой они принимают участие. Придерживаясь целеориентированного подхода, они должны стремиться представлять поведение и информацию в понятном и полезном виде, который поддерживает маркетинговые цели организации и эмоциональные цели персонажей.

Разумеется, визуальный дизайн пользовательских интерфейсов не исключает эстетических соображений, но такие соображения не должны выходить за рамки функционального каркаса.

Дизайн интерфейсов сводится к вопросу о том, как оформить и расположить визуальные элементы таким образом, чтобы внятно отразить поведение и представить информацию. Каждый элемент визуальной композиции имеет ряд свойств, и сочетание этих свойств придает элементу смысл. Пользователь получает возможность разобраться в интерфейсе благодаря различным способам приложения этих свойств к каждому из элементов интерфейса. В тех случаях, когда два объекта обладают общими свойствами, пользователь предположит, что эти объекты связаны или похожи. Когда пользователи видят, что свойства отличаются, они предполагают, что объекты не связаны.

Создавая пользовательский интерфейс, проанализируйте перечисленные ниже визуальные свойства каждого элемента или группы элементов. Чтобы создать полезный и привлекательный пользовательский интерфейс, следует тщательно поработать с каждым из этих свойств:

* 1. Форма

Форма – главный признак сущности объекта для человека. Мы узнаем объекты по контурам. Если мы увидим на картинке синий ананас, мы его сразу опознаем, потому что мы помним его форму. И лишь потом мы удивимся странному цвету. При этом различение форм требует большей концентрации внимания, чем анализ цвета или размера. Поэтому форма – не лучшее свойство для создания контраста, если требуется привлечь внимание пользователя.

* 1. Размер

Более крупные элементы привлекают больше внимания, особенно если они значительно превосходят размерами окружающие элементы. Люди автоматически упорядочивают объекты по размеру и склонны оценивать их по размеру; если у нас есть текст в четырех размерах, предполагается, что относительная важность текста растет вместе с размером и что полужирный текст более важен, чем текст с нормальным начертанием. Таким образом, размер – полезное свойство для обозначения информационных иерархий.

* 1. Цвет

Цветовые различия быстро привлекают внимание. В некоторых профессиональных областях цвета имеют конкретные значения, и этим можно пользоваться. Чтобы создать эффективную визуальную систему, позволяющую пользователю выявлять сходства и различия объектов, используйте ограниченный набор цветов – эффект радуги перегружает восприятие пользователя и ограничивает возможности по передаче ему информации.

* 1. Текстура

Разумеется, изображенные на экране элементы не обладают настоящей текстурой, но способны создавать ее видимость. Текстура редко бывает полезна для передачи различий или привлечения внимания, поскольку требует значительной концентрации на деталях. И тем не менее текстура может быть важной подсказкой. Засечки и выпуклости на элементах пользовательского интерфейса обычно указывают, что элемент можно перетаскивать, а фаски или тени у кнопки усиливают ощущение, что ее можно нажать.

* 1. Расположение

Расположение – это переменная упорядоченная и выражаемая количественно, а значит, полезная для передачи иерархии. Расположение также может служить средством создания пространственных отношений между объектами на экране и объектами реального мира (например, небо в верхней половине, земля в нижней).

Человеческий мозг – великолепное устройство распознавания образов. Он справляется со шквалом входных данных, выявляя визуальные закономерности и создавая для наблюдаемых нами объектов систему приоритетов. Именно способность зрительной системы человеческого мозга к сборке частей визуального поля в образы на основании визуальных якорей (подсказок) позволяет нам обрабатывать зрительную информацию столь быстро и эффективно. Процесс создания визуального дизайна интерфейса должен опираться на наши природные способности к обработке визуальной информации, чтобы обеспечить передачу пользователям информации и отражение возможностей и функций программы.

# ЛЕКЦИЯ КОМПОЗИЦИЯ. ВИД И ОЩУЩЕНИЯ ПРИЛОЖЕНИЯ.

## 8.1 Композиция

Композиция – это сложение, сопоставление, приведение частей в единство. под термином понимается объединение отдельных элементов изображения в единое художественное целое, выразительно раскрывающее идею произведения. Композиция в веб-дизайне – это гармоничное расположение всех элементов сайта на экране таким образом, чтобы максимально раскрывалось их содержание.

Задачи композиции:

1. Выразительно изображать характерные черты объекта (создание акцентов, цветовых и световых контрастов);
2. Достоверно отображать конкретное событие или сюжет;
3. Заинтересовать зрителя, привлечь его внимание изобразительными приёмами.

Цель композиции – сделать так, чтобы все элементы дизайна воспринимались целостно и составляли единую картину. Для этого дизайнер помещает самые важные объекты в центр внимания пользователя. Добиться этого можно с помощью ракурса, цвета, контраста и других техник.

Можно выделить три «закона» композиции:

1. *Целостность* – изображение или предмет целиком охватываются взглядом, как единое целое, не распадаясь на самостоятельные части; также подразумевается устранение лишних подробностей, отвлекающих внимание от наиболее важных элементов изображения;
2. *Наличие доминанты* – подчинённость второстепенного главному; элементы периферийного положения, создающие окружающую среду, способствуют усилению главной идеи; центр композиции не обязательно связан с геометрическим центром изображения;
3. *Уравновешенность* – определяется соотношением площадей цветовых и тональных масс, расположенных в правой и левой части изображения по горизонтали или по вертикали.

Плохая композиция сразу обращает на себя внимание, режет глаз и вызывает раздражение. Хорошее же остаётся в тени, человек не обращает внимания на дизайн, а просто сконцентрирован на содержании.

## 8.2 Основные элементы композиции в веб-дизайне

Цвет являет одним из наиболее очевидных элементов дизайна (как для пользователя, так и для дизайнера). Он может быть изолирован в качестве фона или применяться к другим элементам., таким как линии, фигуры и т. д. С помощью цвета можно привлечь внимание к определённой части сайта. Существуют основные цвета (КЖС/RYB), вторичные цвета (ОЗФ/OGP) и дополнительные цвета (противоположные цвета на цветовом круге). Цвета в интерфейсах обычно подбираются исходя из цветовых схем:

1. *Монохроматические* – один цвет и все его оттенки, тональности и тени. Хорошо сочетаются, создавая успокаивающих эффект.
2. *Аналоговые* – создаются из смежных цветов. Один как доминирующий, другие просто обогащают схему.
3. *Дополнительные* – в основе лежат два цвета с сильным контрастом. Используется для привлечения внимания пользователей. Можно использовать различные оттенки. Используются тёплые и холодные цвета, что обеспечивает контраст;
4. *Триодическая* – создаётся при выборе одного цвета и добавлении в схему двух других цветов, которые должны лежать на одинаковом расстоянии друг от друга на цветовом круге. Преобладающая температура выбирается для переднего плана.
5. *Тетрадические* – оттенок/тональность/тень и придерживаться выбранного повсеместно. По сути смесь из двух перпендикулярных дополнительных схем;
6. *Естественная* – лучшие цветовые комбинации встречаются в природе.

## 8.3 Гармония цвета

*Цветовая гармония* – сочетание отдельных цветов или цветовых множеств, образующих органическое целое и вызывающих эстетическое переживание.

Правила:

1. Ограничение числа цветов имеет много общего с балансом;
2. Наиболее приятная цветовая гамма (а если комфортная скорость – 150?);
3. 60–30–10 – обеспечение баланса цветов в любом пространстве;
4. Сначала чёрно-белый вариант;
5. Не использовать чёрные цвета;
6. Подчеркнуть важность, используя цветовой контраст;
7. Использовать цвет как инструмент влияния на эмоции;

## 8.4 Типографика

*Типографика* – одна из самых важных основ дизайна. Любой шрифт сам по себе уже есть законченный элемент дизайна. Использовать от 1 до 3 шрифтов.

1. Базовый шрифт – основной для материалов сайта;
2. Акцедентный шрифт – для заголовков.
3. Дополнительные – для меню навигации, блоков выделения, мелкого текста.

Дизайнер должен спланировать единую общую схему размеров, отступов/заступов для всех элементов на странице, иерархию заголовков и навигационных элементов.

Четыре типа шрифтов:

* Шрифт с засечками (`serif`);
* Шрифты без засечек (`Arial, sans-serif`);
* Рукописный или курсив;
* Специальные (включая моноширинный).

Некоторые правила выбора шрифта:

1. Шрифт соответствует содержанию и уместен;
2. Размер шрифта как правило от 12 до 16 px;
3. Использовать правильно теги важности;
4. Длина строки от 40 до 80 символов;
5. Правильная высота строки;
6. Коэффициент контрастности максимальный для читаемости (>4.5 для мелкого текста и >3 для крупного текста).

# ЛЕКЦИЯ УДОБСТВО ТЕСТИРОВАНИЯ

## 9.1 Введение

*Юзабилити тестирования* – исследования выполняемые с целью определения, удобен ли некоторый искусственный объект для его предполагаемого применения, основанное на привлечении пользователей в качестве тестировщиков и суммировании полученных от них выводов.

В ходе юзабилити тестирования изучается насколько хорошо пользователи выполняют конкретные стандартные задачи и с какими проблемами они при этом сталкиваются.

Тестирование позволяет:

* Понять, насколько хорошо или плохо работает интерфейс, что может либо побудить улучшить его, либо, если он достаточно хорош – успокоиться.
* Сравнить качество старого и нового интерфейса и тем самым дать обоснование изменениям или внедрениям.
* Найти и опознать проблематичные стороны интерфейса, а при достаточном наборе выборки, а также оценить их частично.

Юзабилити проводится на всех этапах жизни программного продукта (ПП). В процессе тестирования необходимо соблюдать комплексные методы, поэтому тестируют все три основные области:

1. Информационную архитектуру, в которой должны быть:

* интуитивно-понятная структура информации;
* удобные инструменты навигации и вызовы функций приложения;
* наглядный способ представления результатов действия.

1. Рабочие процессы и взаимодействия, здесь проверяются:

* логика использования и логика рабочих процессов;
* эффективное использование приложения, идентификация и устранение избыточных операций;
* простота использования, интуитивное понимание и скорость адаптации пользователя;
* время отклика на действия пользователя, короткие, беспрепятственные пути пользователя;
* плавный процесс выполнения сценариев, корректная обработка ошибок, отказоустойчивость приложения.

1. Графический дизайн. Проверяется:

* простота восприятия и усвоение проверяемой информации;
* целостность и единство представления используемых функциональных и графических элементов в рамках всего приложения.

Для выявления проблем удобства использования приложения, в том числе на этапах планирования и ранних этапах планирования и разработки ПО, используется методика двойной проверки:

* изучение опытов взаимодействия пользователя с приложением, через имитацию поведения пользователей;
* Проверка соответствия, признаки обеспечения удобства пользования и корректного визуального представления в контексте функциональных требований посредством экспертной оценки.

Экспертная оценка приложения осуществляется в соответствии с целями проекта, функциональными и нефункциональными требованиями ПО.

Включают в себя:

* выявление возможных сценариев использования и путей пользователя в контексте бизнес-целей и функционалом приложения;
* анализ информационной структуры приложения;
* анализ интерфейса и элементов интерфейса;
* анализ функционального соответствия.

## 9.2 Имитация поведения пользователя

|  |  |
| --- | --- |
| **Ю тестирование** | **Ю экспертиза** |
| Нужны пользователю | Нужен эксперт |
| Объективные данные | Субъективные данные |
| Три-четыре недели до получения результата | Результат с первого дня |
| На выходе обозначения проблем | На выходе обозначение проблем, плюс вариант решения |

Виды юзабилити тестирования:

1. коридорное тестирование;
2. модерируемое удалённое тестирование;
3. не моделируемое удалённое тестирование;
4. A/B – тестирование. Это автоматизированная проверка двух или более версий одного контента на одинаковых аудиториях.

Крупный недостаток юзабилити тестирования – высокая стоимость. Экспертная оценка является более быстрым и дешёвым способом оценки качества интерфейса.

## 9.3 Этапы юзабилити тестирования

1. Определение проблемы.
2. Формирование гипотез.
3. Определение метрик (справился/не справился, ошибки, отклонения от сценария, понятность сообщений, последовательность действий и время на выполнение заданий).
4. Определение персонажа и сценария теста (определение целевой аудитории и задания тестирования).
5. Подбор респондента. Сначала необходимо определить общие требования к респондентам:

* опыт работы с системой;
* уровень компьютерной грамотности;
* возраст;
* пол;
* уровень эмоциональной открытости респондента.

1. Заполнение анкеты.
2. Вводный инструктаж.
3. Юзабилити тестирование

* скорость работы;
* ошибки;
* проблемы, которые видно сразу (зафиксировать сущность проблемы и текущее время);
* эмоциональные реакции респондента).

1. Опрос респондентов.
2. Анализ результатов.
3. Определение требования для проектирования сайта или формирование рекомендаций для дальнейшего формирования интерфейса.

## 9.3 Особенность разработки интерфейса для автомобильных устройств