

Человеко-машинное взаимодействие

Учебное пособие

Ветров С.В.

ЗабГУ
2022

УДК 004.5
ББК: 32.973
ISBN *****

Рецензенты

Е. А. Михайлова, к.т.н, доцент, Читинский институт (филиал) федерального государственного бюджетного образовательного учреждения высшего образования «Байкальский государственный университет.

Н. В. Пешков, к.ф-м.н., доцент, Забайкальский институт железнодорожного транспорта – филиал Федерального государственного бюджетного образовательного учреждения высшего образования «Иркутский государственный университет путей сообщения» в г. Чите.

Ветров, Сергей Владимирович

Человеко-машинное взаимодействие / С. В. Ветров ; Забайкальский государственный университет. – Чита : ЗабГУ, 2022. – 143 с.

Учебное пособие предназначено для учебно-методической поддержки дисциплины «Человеко-машинное взаимодействие»

Издание адресовано студентам бакалавриата, обучающимся по направлениям 09.03.01 Информатика и вычислительная техника

УДК 004.5
ББК: 32.973
ISBN *****

Оглавление

Введение	5
1 Человек	9
1.1 Внимание	9
1.1.1 Привлечение внимания.	11
1.1.2 Локус внимания.	13
1.1.3 Состояние потока.	13
1.2 Автоматичные задачи	14
1.3 Память	17
1.4 Ментальные модели	21
2 Пользовательский интерфейс	29
2.1 Понятие интерфейса	29
2.2 Классификация видов ПИ	30
2.3 Парадигмы интеффеса	37
2.4 Модальность	39
3 Дизайн и восприятие	43
3.1 Восприятие	44
3.2 Цвет	51
4 Проектирование UX	55
4.1 Проектирование пользовательского интерфейса	55
4.2 Проектирование UX	57
4.2.1 Уровни UX	57
4.2.2 Уровень стратегии	60
4.2.3 Метод персонажей	61
4.2.4 Уровень структуры	67

4.2.5 Уровень компоновки	68
4.2.6 Уровень поверхности	70
4.3 Методология проектирования	72
5 Юзабилити и тестирование	77
5.1 Юзабилити	77
5.2 Юзабилити-тестирование	82
5.2.1 Понятие юзабилити-тестирования	82
5.2.2 А/В-тестирование	83
5.2.3 Тестирование с наблюдением	83
5.2.4 Метод карточной сортировки	85
6 Анализ интерфейса	87
6.1 Количественная оценка пользовательского интерфейса	87
6.1.1 Закон Фиттса	87
6.2 Закон Хика и проблемы выбора	94
6.3 GOMS	96
6.4 Информационная эффективность интерфейса	101
6.4.1 Информативность интерфейса	101
6.4.2 Оценка информативности.	105
7 Типографика и тексты	109
7.1 Классификация шрифтов	110
7.2 Основные понятия из типографики	115
7.2.1 Выбор шрифта	117
7.2.2 Шрифтовая иерархия	120
7.2.3 Выравнивание текста и длина строки.	121
7.3 Текст и синтаксис интерфейса	124
Заключение	133

Введение

В 1969 году в США случилась авария на атомной станции Три-Майл-Айленд. Течение аварии усугубили несколько факторов. Одна единственная сигнализация срабатывала при входе за пределы нормы любого из примерно 100 параметров. Одни из параметров имели критически важное значение, другие нет. Органы управления и индикаторы на пульте станции не были логически сгруппированы. Принтер, печатающий диагностические данные, работал медленно. Во время аварии он отставал на 2 часа.

В 1988 году ракетный крейсер США сбил пассажирский самолёт над Персидским заливом, приняв его за военный. Среди прочих факторов, приведших к инциденту комиссия, расследовавшая случившееся, отметила недостатки в пользовательском интерфейсе. Интерфейс не учитывал в полной мере, что будет использован в сложной боевой обстановке, во время утомительного дежурства. Незаметно переназначенные программой идентификаторы целей привели к катастрофе.

Пользовательские интерфейсы могут быть настолько плохи, что становятся причинами катастрофы. Но гораздо чаще, интерфейсы "всего лишь" снижают продуктивность пользователей, приводят к потере данных, вызывают неудовлетворённость работой, мешают электронной коммерции, делают продукт неконкурентоспособным.

Мы до сих пор сталкиваемся с проблемами в интерфейсах в своих повседневных делах. Читатели, наверняка могут при-

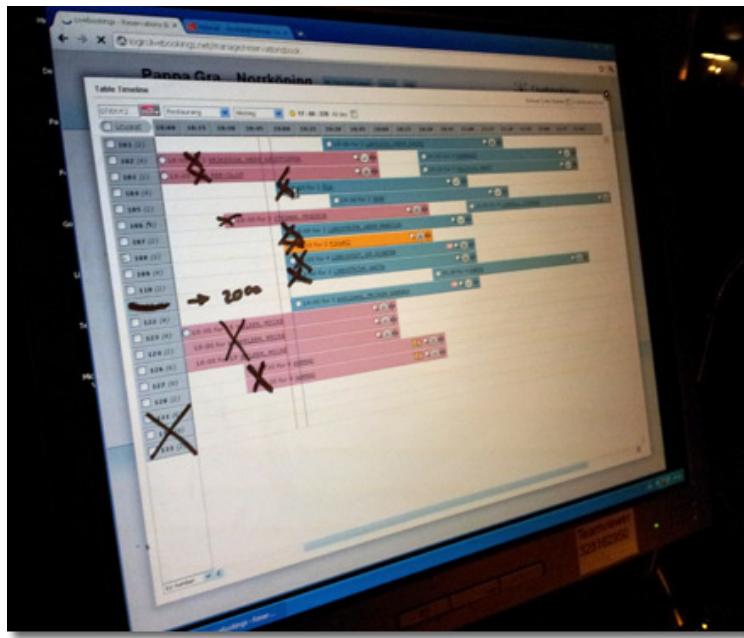


Рисунок 1. Иногда проще рисовать поверх монитора маркером для доски, чем пользоваться программой бронирования столовиков ресторана.

вести много примеров, когда им не удалось совладать с программой, решая несложную задачу, удавалось допустить явные ошибки в работе, которые не были заметны до самого конца или приходилось тратить много времени на рутинную работу, которая, казалось бы, должна быть давно поручена компьютерам.

Одна из основных проблем – непонимание того, что пользователям не нужны программы сами по себе. Пользователям нужны инструменты для решения их собственных задач. Программа должна требовать внимания, времени и сил пользователя только в той степени, в которой это помогает решать проблемы пользователя.

Разработчики программы для ресторана, призванной упростить бронирование столовиков служащими плохо изучили специфику работы главных для себя людей – пользователей их продукта. Работать с программой оказалось проще, если просто делать пометки прямо на дисплее, поверх окна программы (см. рис 1), а не взаимодействовать с программой привычным образом.

Многие пользовательские интерфейсы создание инженерами или программистами несут в себе родовую травму полученную не без участия заказчика продукта. Все эти люди заинтересованы в том, чтобы их детищем пользовались. Но достаточно ли они квалифицированы для проектирования интерфейсов? Хватает ли им здравого смысла, чтобы понять – скорее всего они не компетентны в создании пользовательских интерфейсов. Поэтому так часто пользовательский интерфейс разрабатывается или в последнюю очередь или людьми без достаточной квалификации, не дизайнерами.

Даже если над продуктом работали отличные архитекторы, программисты и тестировщики продукт будет может стать нежизнесспособным, непопулярным и может не оправдать затрат на его создание, если он не подойдёт или не понравится пользователям. Сайт интернет-магазина может *выглядеть* не внушающим доверия, отдельные части сайта могут сбивать некоторых пользователей с толку препятствуя совершению целевых действий, например регистрациям или покупкам (см. пример в параграфе 5.2.2).

Интерфейс – первое с чем сталкивается пользователь. Почти всегда пользователь не может непосредственно оценить внутреннее устройство программы, сколь бы прекрасным оно ни было, но регулярно взаимодействуя с программой он рано или поздно заметит многие недочёты интерфейса.

Современный подход ставит во главу разработку *продукта*¹ для клиента (пользователя) т.е. проектирование всесторонних потребительских качеств продукта.

Этот путь начинается с идеи продукта. Какую задачу он будет решать? Нужно изучить потенциальную целевую аудиторию, чтобы понять их потребности и специфику. Составить набор возможностей продукта, который поможет пользователям

¹Под продуктом будем понимать программу, безразлично для какого устройства, сайт, веб-приложение.

решить их задачи. Придумать, как эти возможности будут реализованы в интерфейсе. Реализовать самый удачный вариант интерфейса, протестировать и исправить недочёты.

К сожалению не существует чётких метрик, по которым можно объективно оценить качество интерфейса, но сравнивая разные варианты можно выбрать лучший. В этом смысле проектирование интерфейсов ещё в большей степени искусство, чем программирование. Поэтому стоит помнить, что любые приведённые правила и рекомендации по проектированию могут иметь исключения.

Вопросы

1. Почему важно уделять внимание интерфейсам?
2. Какие проблемы возникают при проектировании интерфейса?
3. С какими проблемами в пользовательском интерфейсе вы сталкивались?
4. Часто ли программы не позволяли вам упростить рутинную работу?
5. Часто ли вы сталкивались с программами, которые были больше *помощником* чем инструментом?

1 Человек

Руководства по разработке продуктов, взаимодействующих с человеком физически, обычно содержат информацию, основанную на свойствах и возможностях опорно-двигательного аппарата и органов чувств. Совокупность сведений в этой области составляет науку эргономику. На основе этих знаний можно проектировать стулья, столы, клавиатуры или дисплеи, которые с высокой степенью вероятности будут удобны для своих пользователей.

Невозможно эффективно управлять автомобилем, если его органы управления будут значительно удалены друг от друга, требовать существенных физических усилий или экстраординарной точности движений. Так же программа не может быть эффективно использована, если она требует от пользователя беспрецедентной внимательности, способности удерживать в уме большее количество информации и полного понимания внутреннего устройства самой программы.

Важно понимать особенности человеческой психики, его сильные и слабые стороны чтобы проектировать интерфейсы, которые помогут эффективно решать задачи пользователя.

1.1 Внимание

Внимание — избирательная направленность восприятия на тот или иной объект.

Д. Канеман предложил считать внимание ресурсом. Любая

относительно сложная задача или отвлекающий фактор "расходуют" часть внимания человека [4]. Услуга, программа, сайт или устройство редко используются в идеальных условиях, когда им уделено всё внимание пользователя. Они конкурируют за внимание с другими задачами пользователя и с окружающей средой.

Люди часто отвлекаются. Поэтому и интерфейс должен быть таким, чтобы пользователь как можно быстрее мог снова вернуться к работе и допускал как можно меньше ошибок из-за невнимательности.

В том числе поэтому, покупая билет на поезд на сайте железнодорожной компании, мы чаще всего заполняем не все необходимые данные сразу, а поэтапно уделяя внимание вводу данных одной категории за раз: дата поездки, направления и номер поезда; выбор класса обслуживания и места; заполнение данных пассажира; страницы оплаты с обязательным повторением всех данных о поездке. Возможно, придётся отвлечься, чтобы сверится со своим расписанием или написать сообщение друзьям, найти банковскую карту или паспорт. Отвлёкшись на любом из этапов, пользователь должен быстро понять, что он уже сделал и что от него требуется дальше, иметь возможность понять, не ошибся ли он при вводе данных. Такое поэтапное решение задачи ещё и минимизирует использование памяти пользователя, о чём будет сказано ниже.

В лучшем случае пользователь может уделять всё своё внимание решению *своей задачи* с помощью программы, но не пользовательскому интерфейсу. Внимание не только ограничено, но ещё и избирательно. Например, пользователь пишет код в среде разработки, снабжает его комментариями на русском языке, и часто переключая раскладку клавиатуры, ошибается, не обращая внимания на то, какая именно раскладка задействована.

Плохо, если программа может в любой момент показать сообщение о выходе новой версии, об обновлении плагинов или показать несвоевременную подсказку о новых функциях. Тогда программа перетягивает внимание пользователя с решения *его* задачи, но задачу обслуживания самой себя.

1.1.1 Привлечение внимания.

В интерфейсах часто нужно привлечь к отдельным элементам, чаще всего кнопкам. Например в диалоговом окне стоит привлечь внимание к кнопке, которую скорее всего нажмёт пользователь. На сайте привлечь внимание к кнопке целевого действия. Например подписки, регистрации или покупки.

Эффект выскакивания помогает сократить время обнаружения элемента интерфейса [59]. Эффект выскакивания – это независимость скорости поиска целевой стимула (отличного от множества одинаковых) общего количества стимулов. Благодаря этому эффекту человек может быстро находить отличающиеся от остальных объекты на изображении (рис. 1.1). Эффект тем выражение, чем больше отличается искомый объект от остальных. Для статичного изображения большую роль играют цвет, затем размер, форма и наклон объекта.

Если различия целевого стимула и остальных не велики, то эффект выскакивания выражен слабее, время поиска больше зависит от количества остальных визуальных стимулов. Это аргумент в пользу минималистичного дизайна.

Движение – ещё один быстрый способ привлечь внимание пользователя. Однако его можно использовать в относительно узком круге сценариев.

Прерывание опыта – еще один способ привлечь внимание пользователей, особенно когда это делается во время рутинной задачи. Если есть важное объявление или нужно, чтобы пользователи ознакомились с какой-то информацией, то

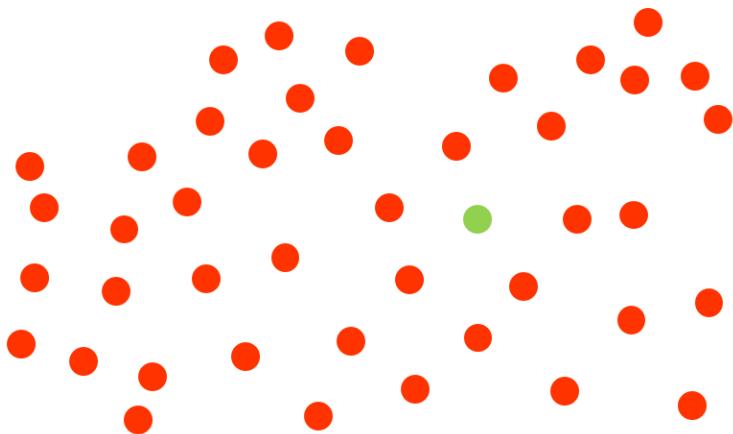


Рисунок 1.1. Благодаря эффекту выскакивания мы быстро и автоматически находим красный круг среди множества других кругов одного цвета.

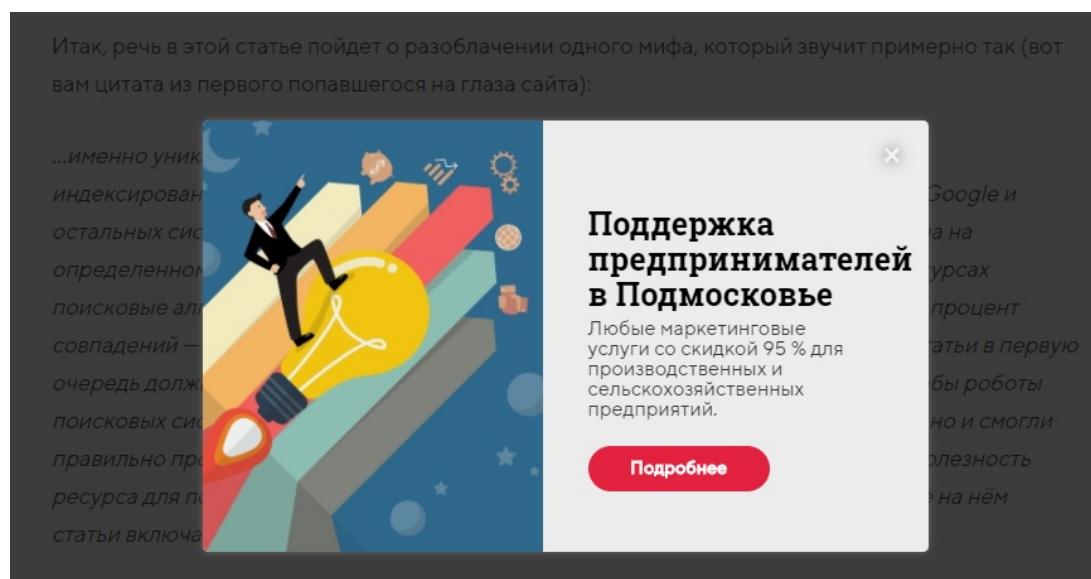


Рисунок 1.2. Всплывающее во время просмотра страницы - это прерывание опыта способ привлечь внимание

отображение этой информации и просьба ознакомиться с ней может быть эффективной техникой (рис. 1.2). Но пользователю обычно потом требуется некоторое время чтобы вернуться к решению своих основных задач, возможно вспомнить то, о чём он думал до всплывающего сообщения. Поэтому этим способом стоит пользоваться с осторожностью. Кроме того это может раздражать пользователей.

1.1.2 Локус внимания.

Идея или предмет, на котором сосредоточено внимание, называют **локусом¹ внимания** (locus of attention) [3].

Локус внимания только один. Поэтому человек может не замечать то, что находится вне локуса внимания. Например, не обращать внимания на раскладку клавиатуры, когда в локусе его внимания находится кодирование алгоритма.

Человек может выполнять несколько задач одновременно, но только одна задача будет выполняться сознательно, она и будет локусом внимания. Выполнение же нескольких дел одновременно – это всегда либо поочерёдная смена локуса внимание либо выполнение других задач не требующих участия сознания (например ходьба). Поэтому невозможно решать несколько интеллектуальных задач одновременно.

Локус может измениться неожиданно. Зазвонил телефон и в локусе внимания уже само устройство или мысль "Кто это звонит?". Возврат в локус внимания прежнего занятия, от которого отвлекли, всегда требует ментальных усилий. Частые переключения с одной задачи на другую снижают нашу продуктивность. А мысленные усилия на такие переключения, создают ложное ощущение, что проделан большой объём интеллектуальной работы.

1.1.3 Состояние потока.

Люди, способные всецело сосредоточиться на некоторой деятельности, забывают о посторонних проблемах и отвлекающих факторах. Такое состояние называется **потоком**.

В состоянии потока человек может быть исключительно продуктивным, особенно если он занят созидательной работой, например конструированием, дизайном, разработкой или на-

¹лат. locus — место — место, область

писания текстов.

Чтобы сделать людей более продуктивными стоит проектировать продукты, вызывающие и поддерживающие состояние потока, и прикладывать все возможные усилия, чтобы избежать любого поведения, потенциально способного разрушить поток. Если программа выбивает пользователя из потока, ему трудно возвращаться в это продуктивное состояние [66].

1.2 Автоматические задачи

Не все решаемые человеком задачи должны быть локусом внимания. Набор текста на клавиатуре слепым методом, так же как и езда на велосипеде или ходьба пешком по тропинке, лучше всего получаются, если человек об этом не задумывается. Но как только задумается, то может сбиться. По мере повторения – или с практикой – выполнение того или иного действия становится привычным, и получается выполнять его не задумываясь.

Любая задача, которую человек научился выполнять без участия сознания, становится *автоматичной*. Любая последовательность действий, которую регулярно выполняют, становится, в конце концов, автоматичной. Автоматизм позволяет выполнять сразу несколько действий одновременно. Все одновременно выполняемые задачи, за исключением не более чем одной, являются автоматичными. Та задача, которая не является автоматичной, является локусом внимания.

Когда человек выполняет одновременно две задачи, ни одна из которых не является автоматичной, эффективность выполнения каждой из них снижается в результате конкуренции за область внимания. Этот феномен психологи называют *интерференцией*.

Чем больший набор задач человек может решать без привлечения сознания – автоматично, тем больше может делать дел

одновременно.

Благодаря способности формировать привычки, относительно несложные действия, которые человек выполняют регулярно, становятся автоматичными. Если пользователь часто вводит один и тот же пароль при входе на сайт или после загрузки ОС, то в определённый момент замечает: вводя пароль легче ошибиться если начать думать о нём во время ввода.

Нужно создавать интерфейсы, которые, во-первых, целенаправленно опираются на человеческую способность к совершению автоматических действий и, во-вторых, развиваются у пользователей такие привычки, которые позволяют упростить ход работы. В случае идеального человекаориентированного интерфейса доля участия самого интерфейса в работе пользователя должна сводиться к формированию автоматических действий. Тогда пользователю легче сконцентрироваться на решении своих задач.

Пользователи запоминают расположение часто используемых кнопок (например как на рис. 1.3, элементов меню и могут ими пользоваться практически не читая надписи на этих кнопках и пунктах меню. Если поменять местами расположение таких часто используемых кнопок (например кнопки "Ок" и "Отмена" в привычном диалоговом окне), то какое-то время человек не сможет эффективно пользоваться программой. Так же как бы не смог бы уверено водить автомобиль, у которого педали газа и тормоза поменяли местами. По той же причине упорядочивать элементы меню в программе в зависимости от частоты их использования плохая идея.

С другой стороны, интерфейс может способствовать формированию вредных пользовательских привычек. Злоупотребление всплывающими окнами с сообщениями (только с кнопкой ОК) или с простым выбором (ОК, Отмена) привело к тому, что многие пользователи привыкли эти окна тут же закры-

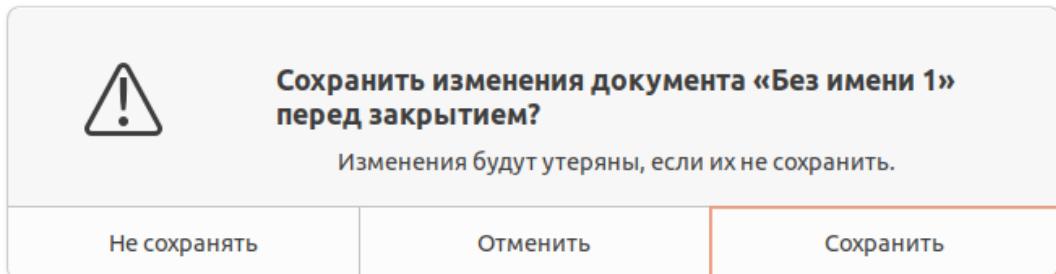


Рисунок 1.3. Если дизайн всех диалоговых окон содержит одинаковый порядок кнопок, то пользователю будет легко сформировать привычку нажимать на нужную кнопку практически не глядя

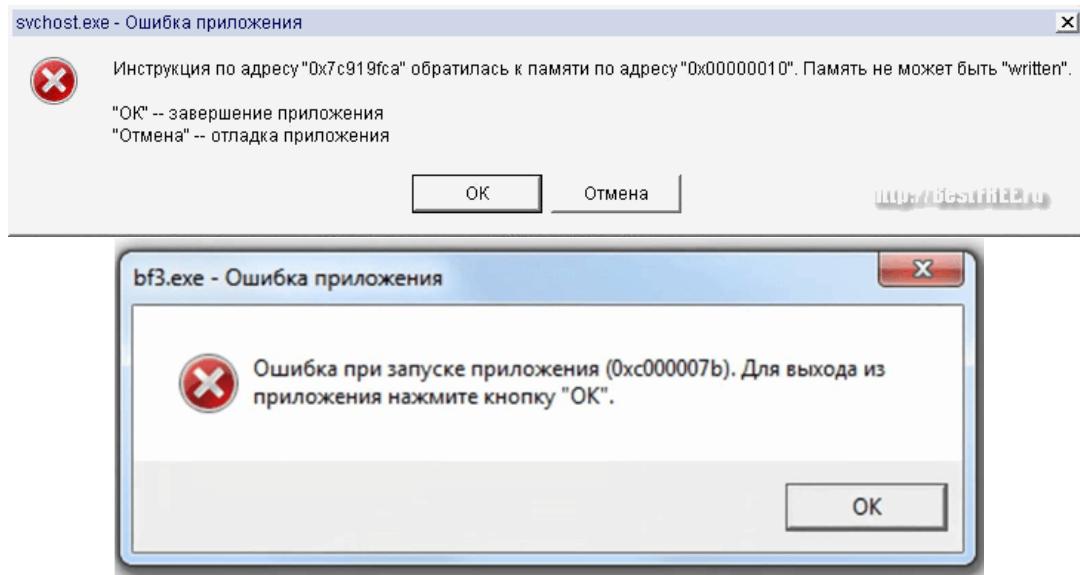


Рисунок 1.4. Злоупотребление диалоговыми окнами с бесполезными для пользователя сообщениями часто приводит к формированию привычки закрывать любые диалоговые окна не вникая в содержимое

вать. Такую привычку легко понять. Чаще всего окно с сообщением не содержит полезной информации (рис. 1.4), например: "ошибка 50" или "неожиданная ошибка". Иногда окно не содержит важной информации и пользователь может просто продолжить работать дальше даже не читая сообщения.

Многие проблемы, которые делают программные продукты сложными и неудобными в использовании, происходят из-за того, что в используемом интерфейсе «человек-машина» не учитываются полезные и вредные свойства человеческой способности формировать привычки. Тенденция предусматривать сразу несколько путей решения одной и той же задачи в одних и тех же условиях как раз мешает формированию привычек.

Множество вариантов действия приводит к смещению локуса внимания пользователя с самой задачи на выбор пути её решения.

Привычки трудно менять. Поэтому человеку бывает трудно, переключится на использование новой клавиатуры или новой, сильно изменившей интерфейс, программы. Это может удержать пользователя от перехода на продукт конкурентов, или наоборот помочь перейти на ваш, если старые привычки пользователей окажутся учтены в новой программе. Многие программисты имеют привычку нажимать комбинацию клавиш Ctrl+S для сохранения файла исходного кода после каждого логически завершённого изменения. Абсолютное большинство программ интерпретируют эту комбинацию клавиш одинаковым образом.

Другая польза автоматических задач – они не отвлекают для пользователя и он может выполнять несложные манипуляции не отвлекаясь на интерфейс программы.

1.3 Память

Человеческую память можно разделить на два вида: долговременную и кратковременную [55].

Кратковременная память — компонент памяти, в который информация поступает из сенсорной памяти, после обработки процессами восприятия, и из долговременной памяти (воспоминания), позволяющий удерживать на короткое время небольшое количество информации в состоянии, пригодном для непосредственного использования сознанием.

Эксперимента Джорджа Миллера показали, что кратковременная память человека способна запоминать в среднем восемь десятичных цифр, девять двоичных цифр, семь букв алфавита и пять односложных слов [56]. Все запоминаемые сущности не были связаны друг с другом по смыслу. Выявленная

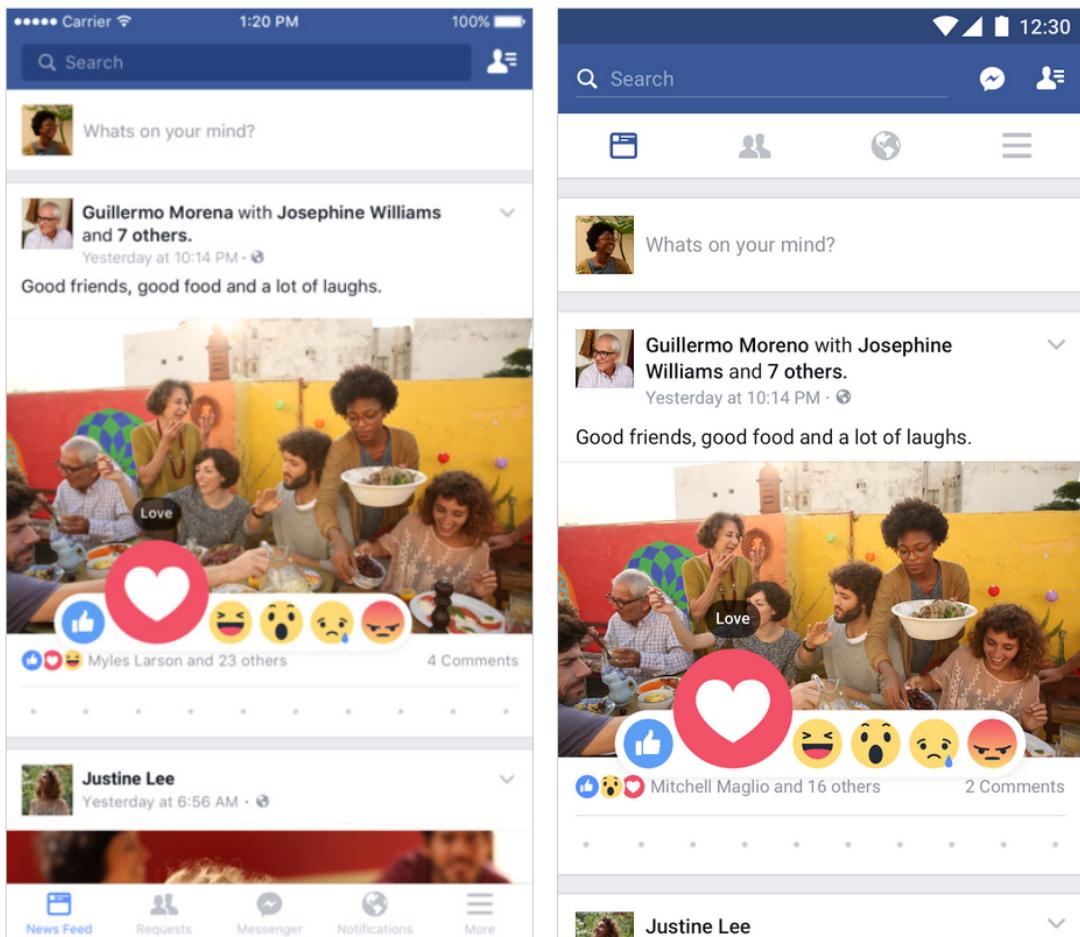


Рисунок 1.5. Приложение Фейсбук для iOS и Андроид подстраивается под привычки пользователей этих операционных систем располагая панель с вкладками так, как принято в данной ОС.

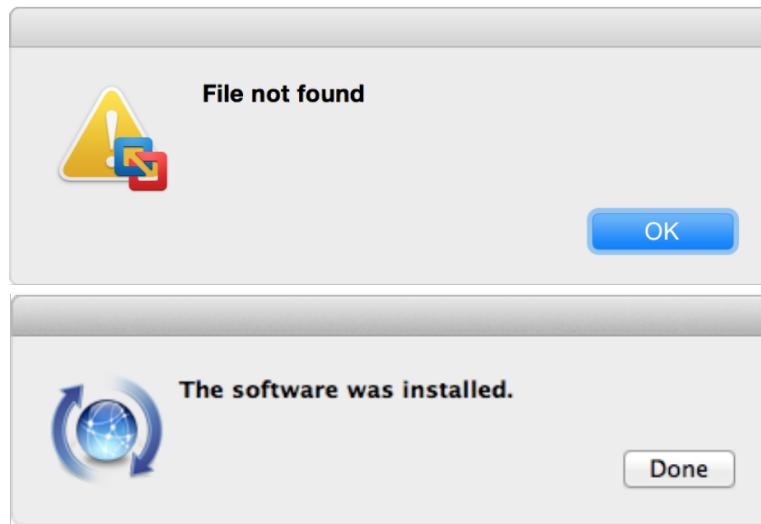


Рисунок 1.6. Программа не должна заставлять пользователя держать в голове информацию, если она сама может её легко хранить и отображать. Если человек отвлёкся, то ему придётся вспоминать о каком файле (верхнее окно) и о какой программе (нижнее окно) ему сообщают [7]

закономерность получила название "Магическое число семь плюс минус два".

Это правило широко освещается в интернет-публикациях по проектированию интерфейсов. Однако опыт показывает прямое применение этого правила, например для скрытия редко используемых пунктов меню, ограничения числа элементов в списках и т.п. не приводит к повышению продуктивности пользователя [7, 58].

Продолжительность хранения информации (при условии, что нет повторения) около 20 сек. После 30 сек. след информации становится настолько хрупким, что даже минимальная *интерференция* разрушает его. Это ещё одна причины учитывать смену локуса внимания пользователя во время использования программы. Чтобы вернуться к работе пользователю нужно заново погрузиться в решаемую задачу (рис. 1.6), желательно как можно быстрее и с меньшими усилиями.

Сохранение не до конца оформленного заказа в интернет-магазине, недописанного сообщения в мессенджере или открытие текстового документа сразу в месте последнего редактиро-



Рисунок 1.7. Сначала данные. программа не должна заставлять пользователя держать в голове данные, вынуждая сначала задать настройки, выбрать режим (программа справа) или фильтры. На изображении справа пользователь сначала вводит число, а потом задает исходные единицы измерения [7]

вания помогает пользователю быстрее перейти к работе.

Если пользователь хочет узнать как 35139 рублей выражаются в долларах США, то плохая программа заставит пользователя сделать много подготовительных действий и держать всё это время длинную цифру в голове прежде чем записать в поле ввода. Например программа конвертации единиц измерения и валют сначала предлагает выбрать вид конвертируемых единиц (длина, масса, объём, валюта), потом выбрать исходную валюту (рубли из большого списка валют), выбрать целевую валюту (снова из большого списка) и только потом предложить записать значение.

Программа не должна заставлять пользователя держать в голове слишком много информации. С учётом того, что пользователь и так в данный момент может совершать сложную интеллектуальную работу. Это же и касается случаев, когда пользователю нужно сделать выбор – удержать в голове большое число вариантов довольно трудно (смотрите также параграф 6.2 о законе Хика). Поэтому в отдельных случаях можно либо сразу делать выбор за пользователя либо предлагать ему небольшое количество готовых вариантов.



Рисунок 1.8. В некоторых случаях ментальные модели могут появляться быстро: как разблокировать телефон? В других случаях ментальные модели могут возникать в процессе активного изучения устройства: Как настроить будильник на 9:00 в этих наручных часах?

1.4 Ментальные модели

Ментальные модели² представляют понимание человека как нечто функционирует [1].

Примеры ментальных моделей: способ завязывания шнурков, способ завести автомобиль; понимание того, что будет если зажать кнопку выключения телефона; местонахождение ближайшей кофейни; способ использования дрели.

Человек учится использовать объект, основываясь на наблюдениях и опыте использования (рис. 1.11). Мы строим ментальные модели основываясь на объекте (иногда достаточно просто посмотреть на него), читая руководство пользователя или спрашивая других людей.

Разные люди имеют разные ментальные модели одних и тех же объектов – то есть люди имеют ментальные модели с разным уровнем абстракции. Кто-то понимает как работает лазерная мышь, как устройство отслеживает изменение положения и как передаёт эти данные. Но у большинства людей просто понимают только общий принцип работы устройства.

²Модуль вообще – это упрощённое представление действительного объекта и/или протекающих в нём процессов

1. Download

2. [Download](#)

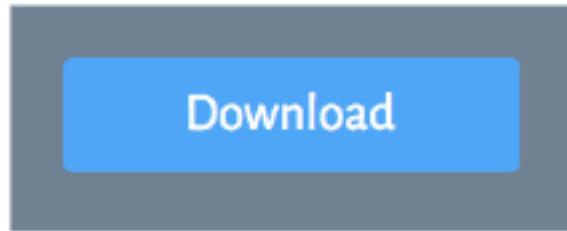


Рисунок 1.9. Какие ментальные модели относительно этих трёх элементов у вас возникают?

Ментальные модели могут быть ошибочными: земля плоская, нельзя выключать компьютер нажав на кнопку включения на системном блоке.

Ментальные модели человека могут быть ошибочными, но при этом работать (рис. 1.10). Если представлять, что внутри фотоаппарата сидит чёртик, способный рисовать увиденное за миллисекунды, то это не обязательно помешает фотоаппаратом пользоваться.

Модель реализации описывает *фактическое устройство* чего либо.

Задача дизайнера интерфейса придумать принципы взаимодействия пользователя и программы. Создать ментальные модели более простые чем модель реализации, но при этом обеспечивающие эффективное взаимодействие пользователя и программы. Таким модели, предлагаемые дизайнером, называются **моделями представления** или моделями дизайнера.

Модели модели могут и не должны не объяснять все особенности устройства или программы, а быть отдельный слоем абстракции – принцип сокрытия сложности. Можно создавать программы, решающие сложные задачи, сколько угодно слож-



Рисунок 1.10. Модель реализации и ментальная модель

но устроенные внутри, но с достаточно простым интерфейсом.

Модели представления должны быть логичными, применяться последовательно и содержать как можно меньше исключений чтобы пользователю было легче их понять. Иконка программы на главном экране телефона воспринимается как программа потому, что анимация открытия этой программы показывает как она как бы вырастает из этой иконки. Значит пользователь может предположить, что иконка обладает свойствами программы (хотя бы частично) и через неё, можно неким образом удалить программу из устройства. Долгое нажатие на иконку открывает меню действий: удалить иконку, удалить приложение, открыть экран "О приложении". Значит такое же действие можно совершить, например с любым элементов на главном экране телефона, например виджетом который показывает время или даже попробовать открыть меню так же зажав имя контакта в телефонной книге.

Непоследовательно использование терминов и обозначений в программе вносит путаницу, требует от пользователя дер-

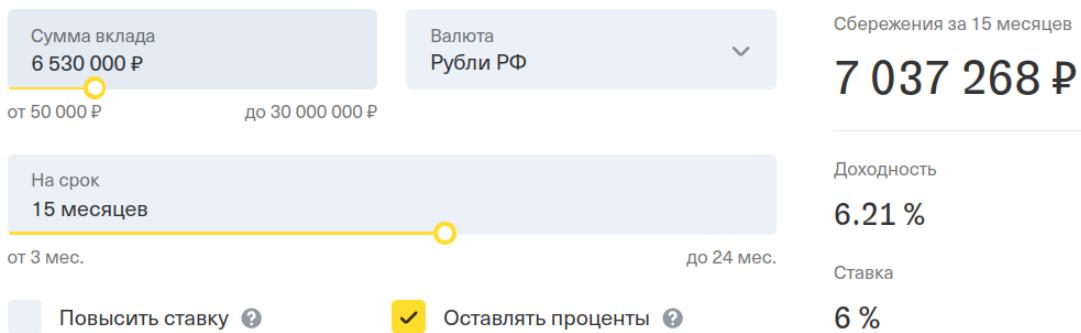


Рисунок 1.11. Вместо сложной таблицы, где перечислены все возможные ставки и сроки вклада на сайте банка есть интерактивный калькулятор. С ним легко экспериментировать меняя ползунками срок и суммы вклада, итоговая сумма рассчитывается автоматически при изменении параметров. Это помогает пользователю понять как устроен вклад - т.е. сформировать ментальную модель

жать в голове дополнительную информацию. Например использование одного и того же значка лупы для обозначения разных действий: открытие окна поиска и изменения масштаба; использование значка в виде креста для удаления данных и для отмены действия.

В хорошем интерфейсе модель представления легко понять или она уже соответствует ментальной модели пользователя. Программы обычно имеют довольно сложное устройство, и принципы их построения скорее всего не знакомы пользователю. Поэтому, модель представления не должна полностью повторять модель реализации.

Проклятие знания – когнитивное искажение, более информированным людям чрезвычайно сложно рассматривать какую-либо проблему с точки зрения менее информированных.

Разработчику программы трудно представить, какие затруднения может вызывать программа у пользователя. Ведь разработчик не только понимает, как взаимодействовать с программой, но и как программа устроена изнутри. Это одна из причин, невысокого мнения программистов о пользователях. Это мешает программисту стать на место пользователя, а значит спроектировать хороший интерфейс.

Ментальные модели редко образуются после чтения инструкции из-за **парадокса активного пользователя**. Парадокс впервые был представлен в исследовании Мэри Бет Россон и Джона Кэрролла, исследователями из IBM, в 1980-х годах. Наблюдая за новыми пользователями, было определено неожиданное на тот момент поведение: пользователи редко читали руководство по продукту. Вместо этого пользователь начинал взаимодействовать с продуктом, чтобы опробовать его, даже если это означало столкновение с ошибками, которых можно избежать, прочитав инструкцию [64].

Парадокс заключается в том, что пользователи могли бы получить выгоду в долгосрочной перспективе сперва разобравшись как пользоваться программой (например изучив руководство, в том или ином виде). Это могло бы оградить их от ошибок и впустую потраченного времени в попутке решить свои задачи неправильным способом. Подход: сделать интерфейс не заботясь о его сложности и понятности, а потом объяснить принципы его работы в руководстве пользователя здесь работает плохо.

Из этого следуют важные выводы. Пользователи не всегда поступают рационально. Но желание как можно быстрее приступить к использованию программы понятно – для пользователя конечная цель – решить свои задачи используя программу, а не изучить программу.

Не все программы возможно сделать понятными для новичка без руководства пользователя. Но при проектировании интерфейса стоит учитывать, что многие пользователи предпочтут научится пользоваться программой на ходу.

Разработчику не стоит рассчитывать на то, что он сможет изменить пользователя. Сможет донести до него существенную информацию о внутреннем устройстве программы и о принципах её работы, сделать пользователя внимательным, заста-

вить поступать рационально, держать в памяти все необходимую информацию, быстро и точно выполнять все действия, решая собственные задачи в программе и заниматься её обслуживанием. Но разработчик может спроектировать интерфейс учитывая человеческие слабости: ограниченность и переменчивость внимания, возможно нежелание изучать программу, неправильные ментальные модели.

Программы вместо человека должны держать всё под контролем, хранить необходимые данные, берегать пользователя от совершения ошибок, а если он их совершил, то давать возможность исправить минимальными усилиями. При решении сложных задач, программа должна помогать пользователю освоить новые ментальные модели и полагаться на имеющиеся, использовать способность человека к формированию привычек для ускорения взаимодействия с программой.

Вопросы

1. Что такое локус внимания? Что такое интерференция?
2. При каких условиях человек может выполнять несколько задач одновременно?
3. Что такое состояние потока?
4. Охарактеризуйте кратковременную память? Какие выводы о ПИ следуют из этого?
5. Что такое автоматические задачи? Как они формируются?
6. Как можно использовать автоматические задачи?
7. Что такое ментальная модель? Приведите примеры.
8. Приведите примеры появления новых ментальных моделей в интерфейсах программ и устройств. Как они изначально были восприняты? Что можно сказать о них теперь?
9. Что такое модель реализации?

10. Что такое модель представления?
11. Как эти понятия относятся друг к другу?
12. Что такое проклятие знания?
13. Как *должны* соотносится ментальная модель и модель представления?
14. Что такое парадокс активного пользователя?

Литература

- Дональд Норман. Дизайн привычных вещей, обновлённое и дополненное издание, 2018, 384 с.
- Канеман Даниэль, "Думай медленно... решай быстро" 2011, 499 с.

2 Пользовательский интерфейс

2.1 Понятие интерфейса

Интерфейс – граница между двумя функциональными объектами, требования к которой определяются стандартом; совокупность средств, методов и правил взаимодействия (управления, контроля и т. д.) между элементами системы [?].

В случае интерфейса пользователя под элементами системы понимают пользователя и, в общем случае, программно-аппаратный комплекс.

Интерфейс пользователя (ПИ, UI – User Interface) – интерфейс, обеспечивающий передачу информации между пользователем-человеком и программно-аппаратными компонентами компьютерной системы¹.

Далее будет идти речь об интерфейсе исключительно пользовательском. Для краткости под программой будем понимать ОС различных устройств, прикладные программы и сайты.

Как видно из определений главная цель ПИ – передача информации между человеком и программой. Эта передача может быть односторонней. Пассажиры получают информацию с табло со временем отправления и прибытия поездов не могут его поменять. Интерфейс может не передавать никакой информации пользователю, но быть доступным для действия пользователя как кнопка вызова лифта без индикации. ПИ может быть невидимым. Например, интерфейс голосового робота опе-

¹ другой важный вид интерфейса в программной инженерии – программный (API), позволяющий взаимодействовать программам между собой

ратора мобильной связи.

2.2 Классификация видов ПИ

Приведём классификацию видов пользовательского интерфейса.

- Визуальный;
 - Текстовый - Text-based Interface (TUI);
 - Графический (ГПИ или ГИП) - graphical user interface (GUI);
- Тактильный (Haptic or kinesthetic communication);
- Жестовый;
- Голосовой;
- Материальный (осознательный) - tangible user interface (TUI);
- Нейрокомпьютерный интерфейс.

Зачастую, конкретный ПИ является комбинацией нескольких видов интерфейса.

Нейрокомпьютерный интерфейс – система, созданная для прямого обмена информацией между мозгом и электронным устройством (например, компьютером). В односторонних интерфейсах внешние устройства могут либо принимать сигналы от мозга, либо посыпать ему сигналы (например, имитируя сетчатку глаза при восстановлении зрения электронным имплантатом). Двунаправленные интерфейсы позволяют мозгу и внешним устройствам обмениваться информацией в обоих направлениях. В основе нейрокомпьютерного интерфейса часто используется метод биологической обратной связи.

Материальный ПИ (tangible user interface, TUI) – это разновидность интерфейса пользователя, в котором взаимодействие человека с электронными устройствами происходит при помощи материальных предметов и конструкций.

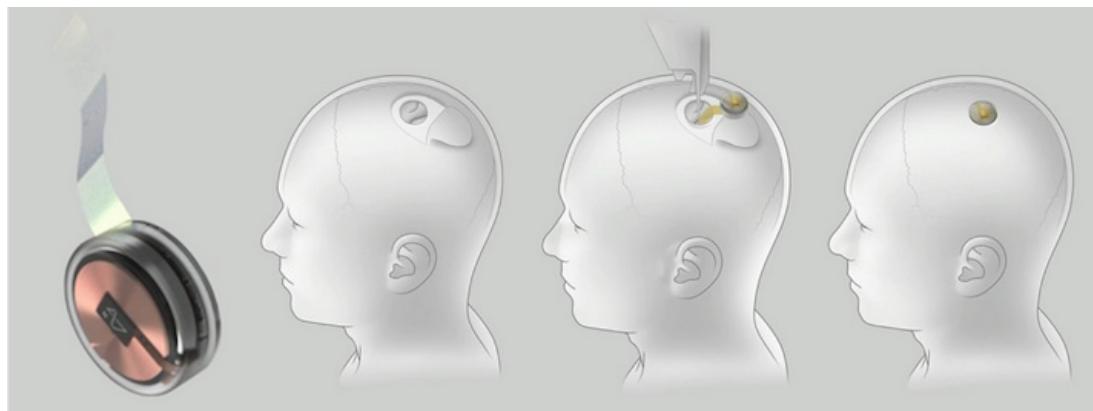


Рисунок 2.1. Имплантируемый нейрокомпьютерный интерфейс Neuralink



Рисунок 2.2. Музыкальный инструмент «reacTable» – электроакустический электронный музыкальный инструмент. Положение и ориентация специальных блоков на интерактивной поверхности влияет на генерируемый аудиосигнал. Демонстрация: youtube.com/watch?v=I9AeUISg-Og



Рисунок 2.3. SandScape – виртуальный ландшафт и его объекты, взаимодействуют с пользователем, формирующим этот ландшафт с помощью песка. Демонстрация: vimeo.com/44538789/

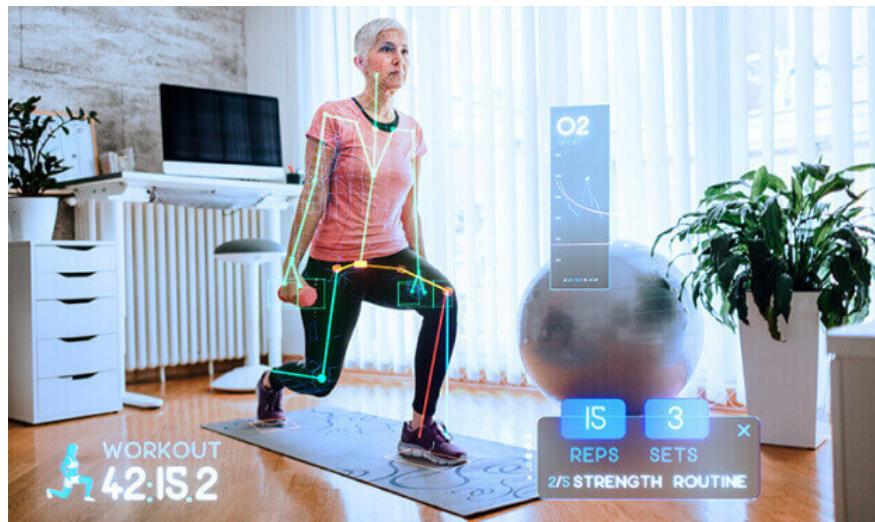


Рисунок 2.4. Azure Kinect DK. Система компьютерного зрения распознаёт положение тела. Демонстрация: youtube.com/watch?v=OoJWmmOlws8

Жестовый ПИ используется в устройствах с сенсорными экранами - мобильных устройствах, интерактивных панелях, где основные жесты – это касание (тап), свайп, поворот и раздвижение двух пальцев. Здесь жестовый ПИ сочетается с графическим интерфейсом. Тачпад или компьютерная мышь – тоже часть жестового ПИ. Компьютерные игры могут использовать устройства захвата движений (PlayStation Move, Wii Remote и др.) или распознавать жесты с видео (Microsoft Kinect, TrackIR, Azure Kinect DK и др.).

Тактильный ПИ предполагает тактильную обратную связь с пользователем. Самый распространённый пример – вибрация игровых контроллеров и телефона.

Текстовый пользовательский интерфейс, (Text user interface, TUI; Character User Interface, CUI) – разновидность интерфейса пользователя, использующая при вводе-выводе и представлении информации исключительно набор буквенно-цифровых символов и символов псевдографики.

Интерфейс командной строки (Command line interface, CLI) разновидность текстового интерфейса (CUI), в котором инструкции компьютеру даются в основном путём ввода с клавиатуры текстовых строк (команд). Также известен под названием кон-



sovashn

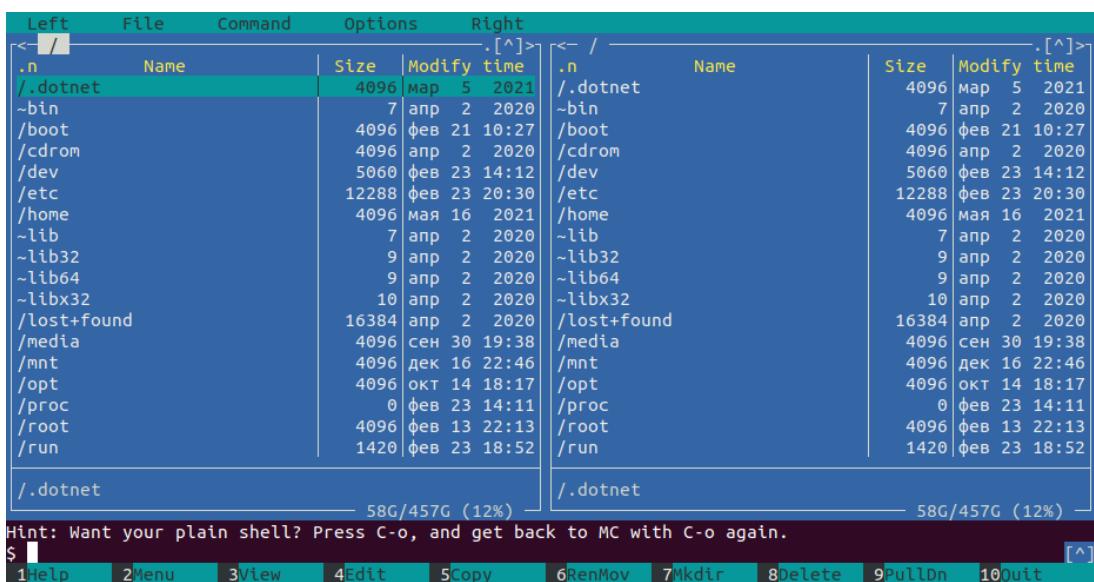


Рисунок 2.5. GNU Midnight Commander – файловый менеджер с текстовым ПИ

```

~ ➤ cd testproject
~/testproject [ master ➤ gco detached-head-state -q
~/testproject ~ fdffaf6 ➤ touch dirty-working-directory
~/testproject ~ fdffaf6± ➤ cd
~ ➤ ssh milly
Welcome to Ubuntu 11.04 (GNU/Linux 2.6.18-308.8.2.el5.028stab101.1 x86_64)
Last login: Wed Sep 26 03:42:49 2012 from 71-215-222-90.mpls.qwest.net
agnoster@milly: ~
Connection to milly.agnoster.net closed.
~ ➤ sudo -s
Password:
$ root@Arya: ~ ➤ top &
[1] 34523
[1] + 34523 suspended (tty output) top
$ rm no-such-file
rm: no-such-file: No such file or directory
x $ rm no-such-file
$ kill %
[1] + 34523 terminated top
$ root@Arya: ~
~ ➤

```

Рисунок 2.6. ZSH - командная оболочка для UNIX, с гибким механизмом автодополнения команд, исправления опечаток, цветовым кодированием состояний репозиториев git

соль.

Такой вид ПИ обладает техническим преимуществами: легко автоматизировать действия, он не требователен к вычислительным ресурсам. Возможность автоматизации можно рассматривать как дополнительную функциональность, которая идёт вместе с интерфейсом. Второе преимущество улучшает субъективный пользовательский опыт на низко производительных устройствах и во время сетевого доступа (например, по SSH) при низкой скорости сети.

С таким типом интерфейса можно выполнять сложные задачи быстрее² чем с ГИП, особенно если учесть возможность автоматического дополнения команд.

Но такой тип интерфейса требует от пользователя многих ментальных моделей – знания команд, из параметров и принципов работы с ними. Эти ментальные модели сложнее формируются (плохая изучаемость, learnability³), в отличие от случая с ГИП, где можно изучить программу рассматривая ПИ и

²для оценки времени на совершение действий см параграф про GOMS

³см. параграф о критериях качества интерфейса

взаимодействуя с ней. Часто здесь нет и хорошей обратной связи – важного атрибута почти любого ПИ.

Но перечисленные недостатки не существенны, для части пользователей. Поэтому такой вид интерфейса широко распространён в программах для семейства ОС Linux. Его наличие ценится многими специалистами. Ещё одно подтверждение тому, что перед созданием продукта важно знать свою целевую аудиторию.

Текстовый ПИ широко распространён в диалоговых системах. Например SMS-запросы, чат-боты для взаимодействия с сервисами заказа услуг, технической поддержки.

Графический пользовательский интерфейс исторически сменил текстовый, но не вытеснил до конца. Здесь нет необходимости знать текстовые команды, название многих утилит их аргументов и параметров. К тому же эти команду нужно вводить с абсолютной точностью.

Количество возможных способов взаимодействия с программой (возьмём все возможные комбинации символов команд и их аргументов в текстовом ПИ) в текстовом интерфейсе огромно. В графическом интерфейсе неделимых способов взаимодействия гораздо меньше. Как правило это клики мышью, прокрутка, клик и перемещение. Клавиатура чаще используется для ввода данных, и реже для ввода команд (горячие клавиши). Благодаря появлению пользовательского интерфейса сократился лексикон взаимодействия (*interaction vocabulary*). При этом элементы интерфейса теперь видны на экране. А диапазон решаемых задач с помощью графического ПИ как минимум не меньше чем для текстового. Далее в этом пособии будет рассматриваться преимущественно графический пользовательский интерфейса.

2.3 Парадигмы интерфейса

Существуют три основных парадигмы интерфейса [10]:

- Метафорическая
- Идиоматическая
- Парадигма реализации

Интерфейсы, построенные согласно парадигме реализации, опираются на понимание того, как работает продукт, то есть основываются на модели реализации, но не на модели представления модель. Интерфейсы могут сочетать в себе все три парадигмы. В самом своём начале ЧМВ как дисциплина в основном полагались на понимание работы программы пользователем. До сих пор многие программы с графическим интерфейсом следуют такому принципу. Они имеют по кнопке на каждую функцию и по диалоговому окну на каждый модуль кода, а их команды и процессы являются точным отражением внутренних структур данных и алгоритмов. Подобные программы просто создавать и тестировать. Но они никак не стремятся помочь пользователю, заставляя его тратить больше времени на изучении программы, а не на решении своих задач в ней. Иногда такой подход приводит к созданию сайтов, который повторяют структуру стоящей за ней организации и её бизнес-процессов.

Метафорические интерфейсы основаны на интуитивных представлениях о работе продукта, то есть на узнавании принципов и элементов заложенных в интерфейс. Такие интерфейсы полагаются на модель представления. Многие понятия и элементы интерфейса в современных программах построены на метафорах: папка, рабочий стол, корзина, переключатель (radio button) и т.д. Яркий пример интерфейса повсюду следующего этому проходу – Microsoft Bob (рис. 2.7) Часто приводится требование к интерфейсу – «интуитивно понятный»,



Рисунок 2.7. Microsoft Bob был выпущен для Windows 3.1. Приложения, встроенные в Bob, представлены соответствующими элементами интерьера: например, при нажатии на часы открывает календарь, а ручки и бумага представляют программу составления писем.

то есть простой в использовании или простой для понимания. Однако это требование ничего не говорит о том, на сколько эффективно с программой может взаимодействовать не новичок.

Дизайн интерфейсов двухтысячных годов широко использовал скевоморфизм. **Скевоморфизм** – это тенденция в дизайне, в основе которой лежит реалистичное изображение объектов (). В скевоморфной графике показан объём предметов: свет, тени, блики и текстуры.

Идеоматические интерфейсы основаны на обучении пользователя тому, как достичь результата, что является для людей естественным процессом. Идиоматическое проектирование, которое Тед Нельсон (Ted Nelson) назвал «проектированием принципов», основано на том, как человек изучает и применяет идиомы и речевые обороты. Идиоматические пользовательские интерфейсы решают проблемы, с которыми не справляются предыдущие два типа интерфейсов, поскольку сосредоточиваются не на технических знаниях и не на интуитивных



Рисунок 2.8. Скевомофизм в iOS: приложение калькулятор подражает калькулятору Braun, созданному Дитером Рамсом для фирмы Braun/

представлениях о функциях, а на изучении простых, неметафорических визуальных и поведенческих идиом, необходимых пользователю для достижения целей и решения задач.

Многие элементы интерфейса – это скорее идиомы чем метафоры: окно, заголовок окна, кнопка закрытия, гиперссылка, выпадающий список. Компьютерная мышь – эта метафора, которая описывает внешний вид устройства но не принцип его использования. Но пользователь легко может понять принцип её работы. Это – идиоматическое освоение.

Идиомы легко запоминаются и их можно понять их контекста. Большая часть того, что мы знаем, усвоена нами без понимания: лица людей, общественные и личные отношения, мелодии, названия брендов, расположение комнат и мебели в нашем доме или офисе. Мы не понимаем, почему чье-то лицо выглядит так, а не иначе; мы просто знаем это лицо.

2.4 Модальность

Жест – действие или последовательность действий, которые человек не разделяет на составляющие, а выполняет как бы

единным движением. Для опытного пользователя ввод короткого слова с клавиатуры — один жест. Для начинающего отдельным жестом будет нажатие каждой клавиши [7].

Один и тот же жест может вызывать разные действия в разных состояниях интерфейса. Кнопка в текстовом редакторе начинает новую строку, а в чате — отправляет сообщение. Педаль газа обычно значит «ехать вперёд», но если включена задняя передача, то уже «ехать назад».

Интерфейс называется модальным, если в нём есть состояния, которые человек не осознаёт во время жеста, но в которых этот жест интерпретируется по-разному.

Ошибка, совершённая человеком из-за того, что он не осознавал, в каком состоянии находился интерфейс, и получил не тот результат жеста, которого ожидал, называется модальной ошибкой.

Модальная ошибка привела к крушению самолёта рейса 214 «Азиана-эйрлайнс» в июле 2013 года. Пилот не осознавал, что в текущем режиме автопилота не работает автомат тяги, и продолжал управлять самолётом, не следя за скоростью.

Режим часто не является локусом внимания пользователя. Поэтому мы часто ошибаемся вводя текст не переключив раскладку. Если режим плохо считывается пользователем, то это приводит к непредсказуемой (с точки зрения пользователя) работе программы и увеличению числа ошибок. Если при проектировании интерфейса нельзя избежать режимов, то стоит сделать их заметными (рис. 2.9)

Квазирежимом Джейф Раскин называет такое состояние интерфейса, в котором пользователь его удерживает физически. Квазирежим невозможно не заметить.

Кнопка Капслок переключает между режимами ввода строчных и прописных букв. Это приводит к модальным ошибкам:

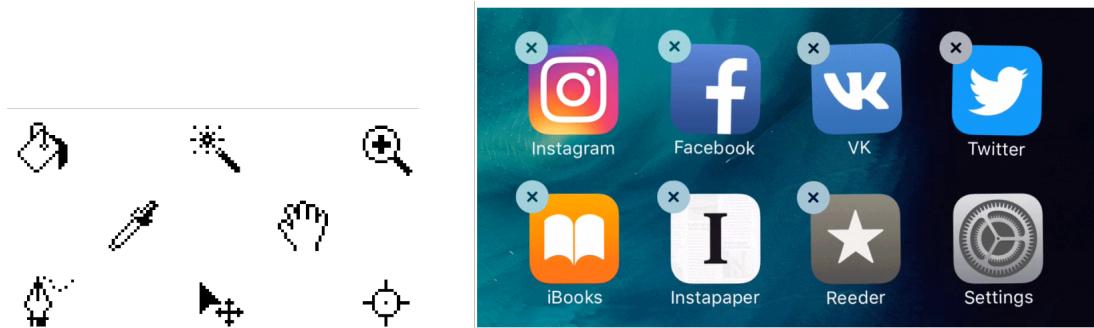


Рисунок 2.9. В графическом редакторе режим определяет форму курсора. В iOS в режиме редактирования домашнего экрана иконки подрагивают [7]

если забыть отключить, по ошибке пишешь большими буквами.

Кнопка Шифт включает квазирежим ввода прописных букв — буквы пишутся прописными лишь пока кнопка зажата. Ввод отдельной прописной с шифтом воспринимается человеком не как работа в другом режиме, а как использование другого жеста. Если же человеку понадобится ввести заглавными целое слово, то он будет постоянно физически ощущать особый режим клавиатуры.

Вопросы

1. Что такое пользовательский интерфейс (ПИ)?
2. Относятся ли к ПИ статичные элементы (текст, изображения) окна приложения или веб-страницы?
3. Приведите классификацию видов ПИ
4. Что такое командный интерфейс?
5. Опишите достоинства и недостатки графического и текстового интерфейса
6. Приведите примеры для каждого вида ПИ
7. Какие парадигмы пользовательского интерфейса вы знаете? Опишите каждую из них

8. Что такое скевоморфизм?

9. Что такое модальность?

10. Что такое квазирежим?

3 Дизайн и восприятие

Точного определения понятия дизайн не существует. Поэтому приведём наиболее общее.

Дизайн – комплексный инструмент создания и оптимизации многосторонних потребительских качеств продукта – изделий, услуг, процессов и среды, – наиболее полно отвечающих потребностям человека и общества [16].

Дизайн часто сравнивают с близкой областью – искусством. Но в отличии от последнего, дизайн всегда призван решать некую задачу.

Стоит подчеркнуть, что дизайн в широком смысле не ограничивается только эстетическими качествами продукта. Например, рекламный баннер должен быть не только эстетичным, но и в понятной форме доносить нужную информацию до человека. Быть заметным и, наконец, выполнить свою основную функцию – заинтересовать потенциального клиента.

Дизайн не обязательно предполагает статичность. Дизайн – это то не то, как предмет выглядит, а то, как он работает¹. Дизайн взаимодействия с пользователем – отдельная отрасль дизайна хороший тому пример. Модель представления – тоже объект дизайна.

Дизайн-система – набор компонентов, правил, предписаний и инструментов для повышения качества и скорости разработки продуктов, а также эффективной поддержки существу-

¹известная цитата Стива Джобса – не определение дизайна, а акцент на отдельный аспект понятия

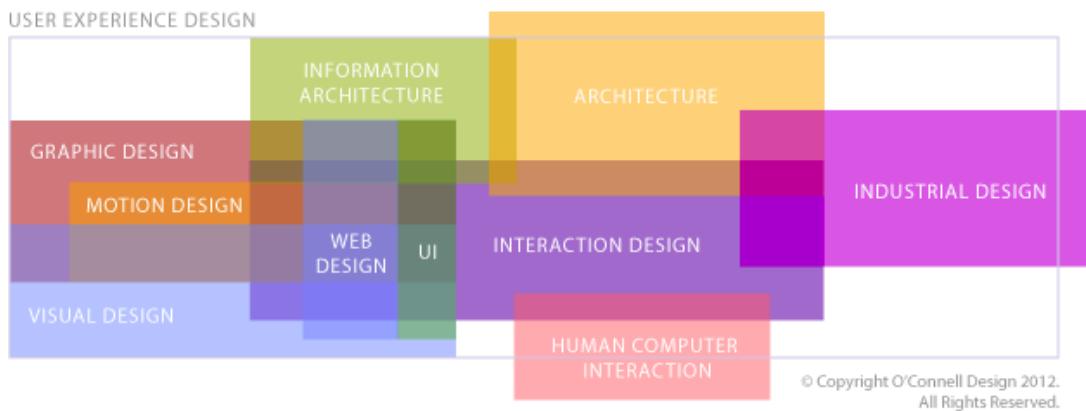


Рисунок 3.1. Отрасли дизайна

ющих.

- Гайдлайны и руководство по стилю
- Фреймворки
- UI-киты, шаблоны
- Наборы UX-паттернов
- Библиотеки готовых компонентов
- Документация, правила, рекомендации

3.1 Восприятие

Как было отмечено запечатление окружающего мира человека условно можно разделить на три уровня:

- ощущение – отражение отдельных свойств и предмета;
- восприятие – целостный образ совокупности свойств предмета;
- представление – сохранившийся в сознании образ предмета, который воспринимался раньше.

Дизайн призван воздействовать на человека на всех трёх уровнях.

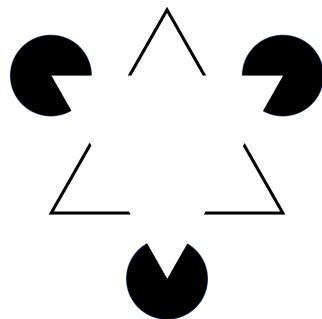


Рисунок 3.2. Треугольник Канижа. На изображении нет четко отчерченного белого треугольника, но человеческое восприятие достраивает его.

Гештальтпсихология (нем. Gestalt – личность, образ, форма) – общепсихологическое направление, связанное с попытками объяснения прежде всего восприятия, мышления и личности. В качестве основного объясняющего принципа гештальтпсихология выдвигает принцип целостности.

Первичными данными психологии являются целостные структуры – гештальты.

Примером противоположной работы восприятия может служить расстройство восприятия – предметная агнозия: человек видит предметы как сумму отдельных частей, но не может составить целостный образ.

Целостность восприятия и его упорядоченность достигаются благодаря следующим принципам:

- близость (Law of Proximity) – стимулы, расположенные рядом, имеют тенденцию восприниматься вместе;
- схожесть (Law of Similarity) – стимулы, схожие по размеру, очертаниям, цвету или форме, имеют тенденцию восприниматься вместе;
- целостность – восприятие имеет тенденцию к упрощению и целостности;
- замкнутость – отражает тенденцию завершать фигуру так, что она приобретает полную форму;

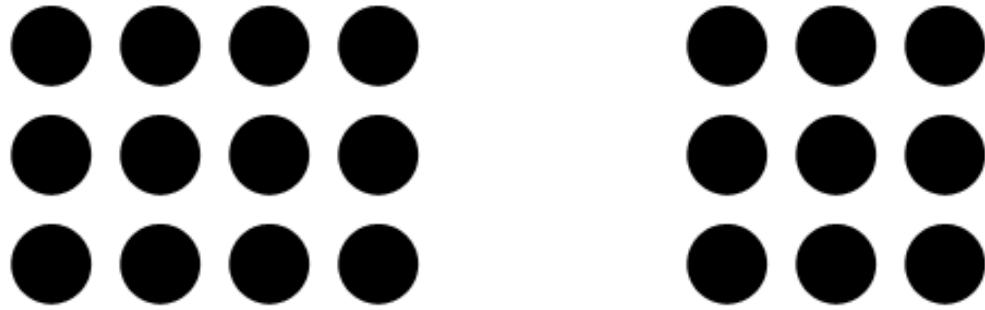


Рисунок 3.3. Принцип близости – близко расположенные объекты воспринимается связанными по смыслу [61]

- смежность – близость стимулов во времени и пространстве. Смежность может предопределять восприятие, когда одно событие вызывает другое;
- общая зона – принципы гештальта формируют наше повседневное восприятие наравне с обучением и прошлым опытом; предвосхищающие мысли и ожидания также активно руководят нашей интерпретацией ощущений.

Принцип близости, называемый ещё теорией близости, возможно один из самых важных принципов восприятия в дизайне. Мы чаще всего неосознанно связываем близко расположенные объекты по смыслу, а удалённые – нет. Очевидный факт: текст расположенный ближе всего к картинке воспринимается как её название или подпись к ней, на самом деле следствие психического закона восприятия.

Порой возникающий спор делать ли подпись к картинке снизу или сверху, в первом приближении имеет простое разрешение в первом приближении – подпись должна быть ближе к своей картинке, чем ко всем остальным.

Если в дизайне есть чёткая и ясная система задающая расположение объектов, то человек её почти наверняка заметит. Поэтому в дизайне широко используются модульные сетки. На-

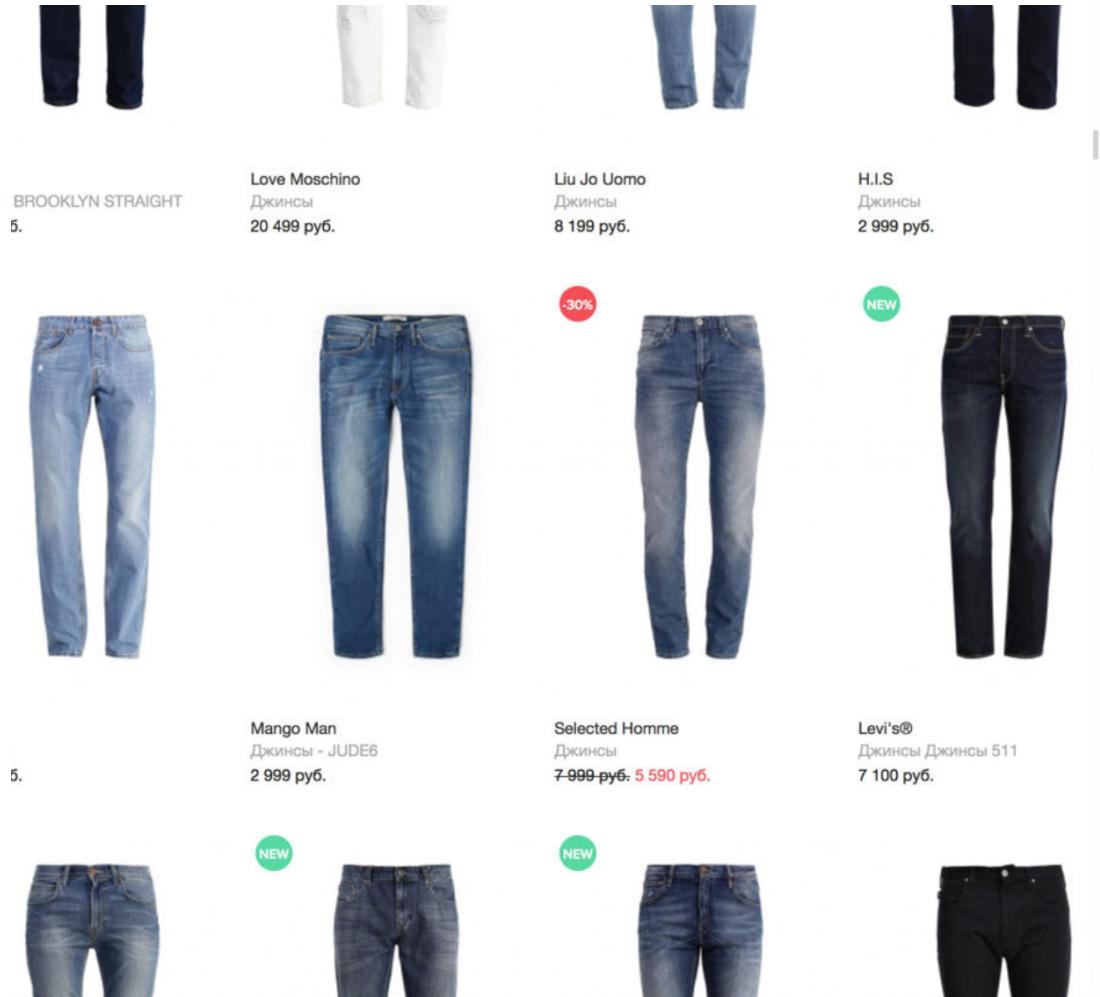


Рисунок 3.4. Дизайнер не использовал принцип близости: нельзя понять к каким изображениями относятся подписи. Нужно сделать подписи ближе к связанным изображениям.

Рисунок 3.5. внутренние расстояния должны быть меньше внешних

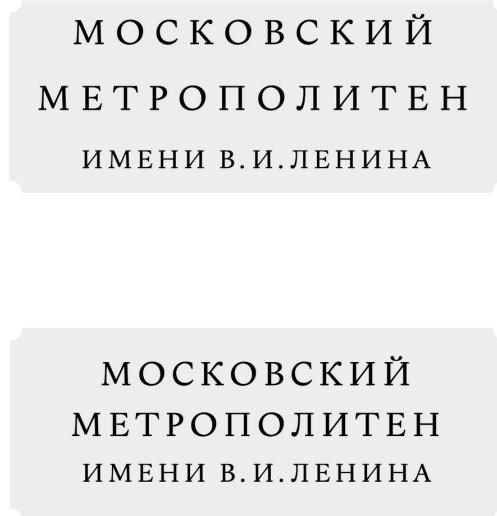


Рисунок 3.6. Верху: принцип "внутреннее \leq внешнее" нарушен: расстояния внутри слова (между буквами) и особенно между словами надписи больше чем между словами и границами области. Текст на нижнем изображении выглядит более целостным, чем на верхем

пример поэтому, в таблицах не всегда нужно изображать линии, разделяющие строки и столбцы, если выравнивание текста говорит само за себя (3.2).

Правило: внутренние расстояния должны быть меньше внешних – следствие закона близости [17].

За счёт использования принципов непрерывности и близости можно избавиться от лишних графических элементов: рисунок 3.9 и рисунок 3.10. Колонки таблицы были выровнены и хорошо определяются без разграничающих линий, отделяемые друг от друга пустотой (расстоянием). Анимация пошагового улучшения таблицы: raw.githubusercontent.com/ivtipm/HCI/master/etc/

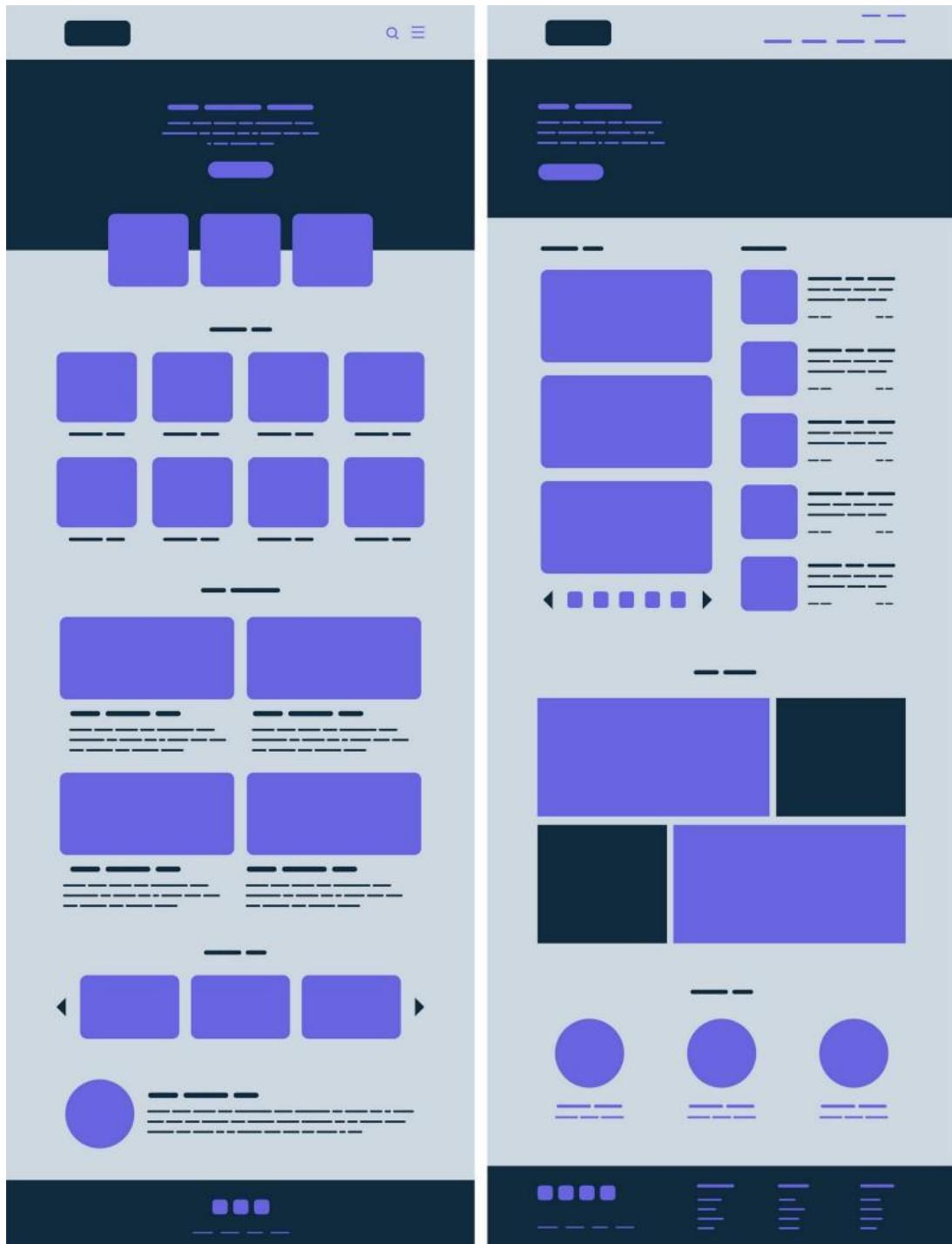


Рисунок 3.7. Принцип близости. Большое значение имеет пустота, воздух – он отделяет объекты друг от друга. Расстояние между отдельными группами элементов – большое, между элементами в группе – меньшее.

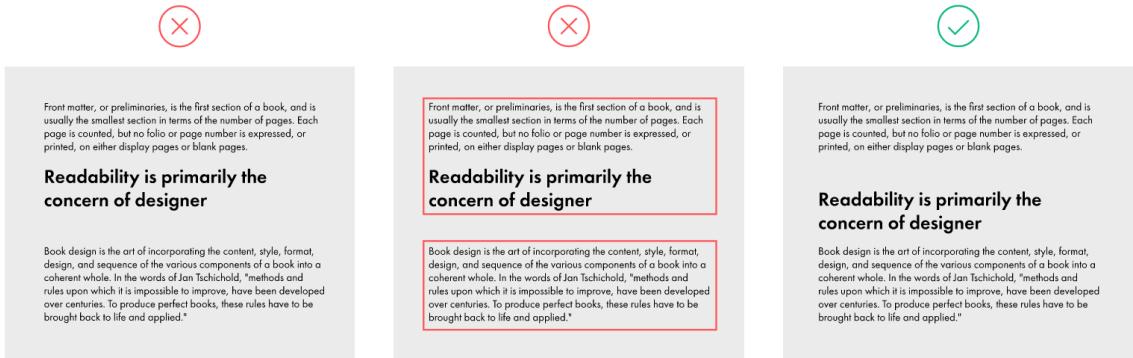


Рисунок 3.8. Частая ошибка: заголовок текста "висит" между абзацами, в то время как должен быть ближе к следующему абзацу, к которому он и относится.

Role	Name	Year of the...	Debut	Number of Fans	Takedown Rate
Face (The Hero)	The Ultimate Warrior	Tiger	May-2011	97320.00	86.2
Face (The Hero)	Hulk Hogan	Oxen	Jan-2008	988551.00	61.978
Face (The Hero)	Macho Man Randy Savage	Monkey	Feb-2008	157618.00	59.29
Face (The Hero)	Hacksaw Jim Duggan	Pig	Mar-2008	30300.00	53.4332
Face (The Hero)	Superfly Jimmy Snuka	Dragon	Mar-2008	12341.00	52.7
Heel (The Bad Guy)	Rowdy Roddy Piper	Rooster	Jun-1968	71645.00	45.4
Heel (The Bad Guy)	The Million Dollar Man Ted DiBiase	Rat	Apr-1975	449342.00	43.7689
Heel (The Bad Guy)	Mr. Perfect Curt Henning	Rat	May-1980	13773.00	38
Heel (The Bad Guy)	Jake the Snake Roberts	Snake	Jul-1975	5609.00	37.99
Jobber (The Unknown)	Brad Smith	Sheep	Aug-2008	1103.00	36.316
Jobber (The Unknown)	Ted Duncan	Sheep	Aug-2008	200.00	33.61
Jobber (The Unknown)	Joey the Uber Nerd Cherdarchuk	Snake	Aug-2008	5.00	21.0196

Рисунок 3.9. Таблица с ошибками в дизайне. Текст выровнен по центру, что приводит к необходимости разделять колонки линиями, что, в свою очередь, увеличивает количество визуального мусора. Числа выровнены не по разрядам.

Role	Name	Year of the...	Debut	Thousands of Fans	Takedown Rate
Face (The Hero)	The Ultimate Warrior	Tiger	May-2011	97.3	86.2
	Hulk Hogan	Oxen	Jan-2008	988.6	62.0
	Macho Man Randy Savage	Monkey	Feb-2008	157.6	59.3
	Hacksaw Jim Duggan	Pig	Mar-2008	30.3	53.4
	Superfly Jimmy Snuka	Dragon	Mar-2008	12.3	52.7
Heel (The Bad Guy)	Rowdy Roddy Piper	Rooster	Jun-1968	71.6	45.4
	The Million Dollar Man Ted DiBiase	Rat	Apr-1975	449.3	43.8
	Mr. Perfect Curt Henning	Rat	May-1980	13.8	38.0
	Jake the Snake Roberts	Snake	Jul-1975	5.6	38.0
Jobber (The Unknown)	Brad Smith	Sheep	Aug-2008	1.1	36.3
	Ted Duncan	Sheep	Aug-2008	0.2	33.6
	Joey the Uber Nerd Cherdarchuk	Snake	Aug-2008	0.0	21.0

Рисунок 3.10. Таблица после исправления ошибок в дизайне. Выравнивание текста по левому краю и пустоты между колонками создают их чёткие границы. Нет нужды в отделении их друг от друга линиями. Равномерный белый фон уменьшает количество визуального мусора.

3.2 Цвет

Модель RGB (red, green, blue — красный, зелёный, синий) или КЗС — аддитивная² цветовая модель, описывающая способ кодирования цвета для цветовоспроизведения с помощью трёх цветов, которые принято называть основными. Выбор основных цветов обусловлен особенностями физиологии восприятия цвета сетчаткой человеческого глаза.

Интенсивность каждого цвета представляется в виде одного октета, значения которого обозначаются для удобства целыми числами от 0 до 255 включительно, где 0 — минимальная, а 255 — максимальная интенсивность. Октеты часто записывают подряд в шестадцатеричной кодировке. Например **#395fb6** представляет соответственно интенсивности красного, зелёного и синего в десятичной системе счисления 57, 95, 182.

Такое представление формально соответствует набору колбочек в глазу человека — клеток, чувствительных к красному зелёному и синему свету. Это же представление широко рас-

²аддитивная модуль — модуль цвета получаются путём добавления к чёрному цвету; отсутствие излучения — нет никакого цвета — чёрный, смешение всех трёх в определённой пропорции — даёт белый

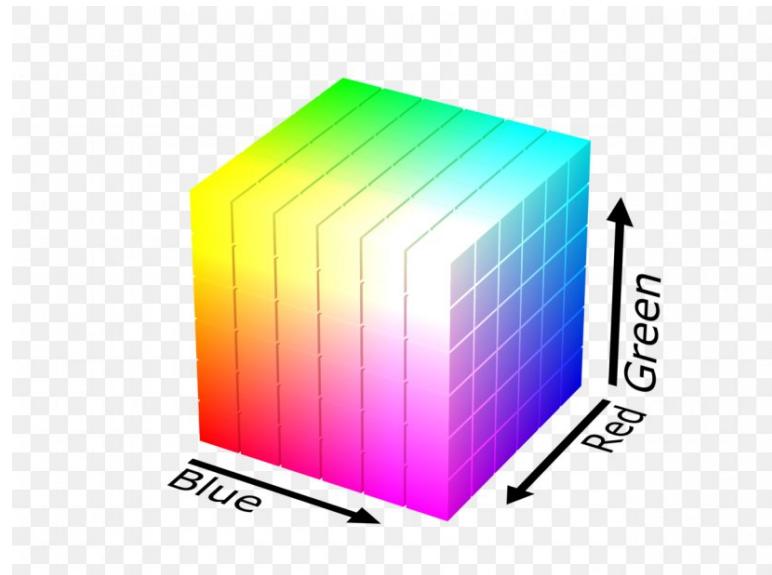


Рисунок 3.11. Три цветовые координаты в модели RGB

пространено в программном обеспечении и устройствах.

Но с точки зрения дизайнера, этот способ кодирования цвета не всегда удобен. Часто приходится изменять насыщенность и яркость конкретного цвета. Но Как сделать цвет `395fb6` светлее? Как сделать этот цвет менее насыщенным? Необходимые изменения трудно выразить в виде изменения интенсивности красного, зелёного и синего.

Модель HSV предназначена решить этот недостаток. HSV (Hue, Saturation, Value — тон, насыщенность, значение) или HSB (англ. Hue, Saturation, Brightness — тон, насыщенность, яркость) — цветовая модель, в которой координатами цвета являются (см. рис 3.12):

- Hue — цветовой тон, (например, красный, зелёный или сине-голубой). Варьируется в пределах $0\text{--}360^\circ$, однако иногда приводится к диапазону $0\text{--}100$ или $0\text{--}1$.
- Saturation — насыщенность. Варьируется в пределах $0\text{--}100$ или $0\text{--}1$. Чем больше этот параметр, тем «чище» цвет, поэтому этот параметр иногда называют чистотой цвета. А чем ближе этот параметр к нулю, тем ближе цвет к нейтральному серому. Value (значение цвета) или Brightness — яркость. Также задаётся в пределах $0\text{--}100$ или $0\text{--}1$.

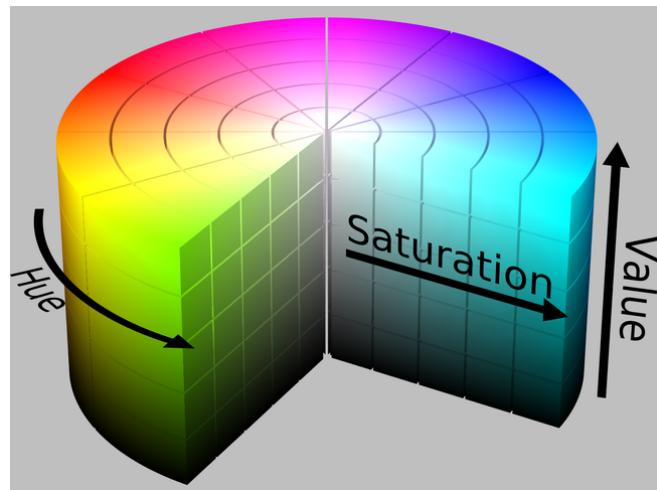


Рисунок 3.12. Модель HSV (HSB). Чтобы изменить только насыщенность (saturation) или только светлоту (value) нужно изменить единственное число в коде цвета.

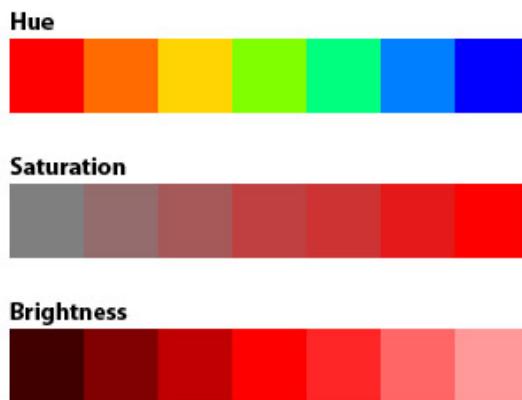


Рисунок 3.13. Пример изменения насыщенности и яркости для одного тона

Модель была создана Элви Рэем Смитом, одним из будущих со-основателей Pixar, в середине 1970-х. Она является нелинейным преобразованием модели RGB.

Цвет 395fb6 в представлении HSV: 222° , 69%, 71%.

Вопросы

1. Что такое дизайн?
2. Что такое ощущение, восприятие и представление? Чем они отличаются?
3. Перечислите принципы целостного восприятия. Опишите

каждый из них.

4. Что такое правило внутреннего и внешнего?
5. Какие существуют модели представления цвета?
6. В чём преимущества цветовых моделей HSV (HSL) перед RGB?

Литература

1. Голомбински, К. Добавь воздуха! Основы визуального дизайна для графики, веба и мультимедиа / К. Голомбински, Р. Хаген. — : Питер, 2013. — 272 с. — Текст : непосредственный.
2. Горбунов А. С. Типографика и вёрстка. — М.: Изд-во Бюро Горбунова, 2015. - 175 с.

4 Проектирование UX

4.1 Проектирование пользовательского интерфейса

Набор элементов интерфейса и правил взаимодействия описывают систему человек-машина лишь частично.

С одним и тем же интерфейсом взаимодействуют пользователи с разными целями. С одним и тем же интерфейсом взаимодействуют пользователи с разным опытом. Пользователи имеют разные ожидания и представления относительно интерфейса. Пользователи по-разному оценивают продукт: кому-то он нравится, кому-то нет. Введём понятие, которое как можно более полно характеризует взаимодействие пользователя и продукта.

Опыт пользователя, опыт взаимодействия (User eXperience, UX) – это восприятие¹ и ответные действия пользователя, возникающие в результате использования и/или предстоящего использования продукции, системы или услуги.

Более лаконичное определение дал Д. Норман. **User Experience** – все аспекты взаимодействия конечного пользователя с компанией, её услугами и продукцией [60].

UX более субъективное понятие, чем *интерфейс пользователя* потому что включает в себя ещё и реакцию пользователя.

Программа может иметь идеальный UI, который позволяет решать задачи пользователя быстро, и эффективно. Но если

¹целостный образ предмета

она чрезвычайно требовательна к ресурсам ПК – это пример плохого UX, при хорошем UI.

Книжный магазин с отличным поиском, аннотациями и удобными способами оплаты. Но небольшой выбор книг – снова плохой UX, пользователь, скорее всего, не получит того, чего желает.

Программа может иметь хороший, удобный и понятный UI, но долго запускаться, неожиданно закрываться, иметь не достаточную (для своей области применения) функциональность, а значит UX может быть не таким же хорошим как UI.

С другой стороны, программа может соответствовать ожиданиям пользователя, вызывать хорошие эмоции при не слишком выверенном UI².

Таким образом, можно создать положительную субъективную оценку от продукта, который имеет относительно серьёзные недостатки.

Нередко хороший или плохой UX становится следствием когнитивных искажений.

Например, эффект IKEA – это когнитивное искажение, которое появляется, когда покупатели непропорционально высоко оценивают значимость (ценность) товаров, которые они создают отчасти сами (например, собирают из деталей). Выполнение длительной операции (запуска программы, скачивание обновления, сохранение настроек) воспринимается пользователем приятнее, если он видит индикацию прогресса. Например, полосу прогресса, изменяющиеся сообщения и т.п.

Некоторые операции, которые выполняются (по мнению пользователей) слишком быстро снабжают более длительной, по сравнению с фактическим временем выполнения, анимацией выполнения. Из-за мгновенного выполнения важной операции,

²как и бренд ≠ качество товара

у пользователя может сложится впечатление, что она не выполнена или выполнена с ошибкой.

Поэтому важно проектировать не только UI, но и учитывать другие аспекты взаимодействия человека и программы.

4.2 Проектирование UX

4.2.1 Уровни UX

Таким образом, задача создания продукта может рассматриваться как задача создания положительного UX. Опыт пользователя плохо формализуется в силу того, что может отличаться от пользователя к пользователю и описывает в том числе субъективную сторону взаимодействия пользователя и продукта. Тем не менее можно выделить общие характеристики если не для всех пользователей, то для большей части целевой аудитории.

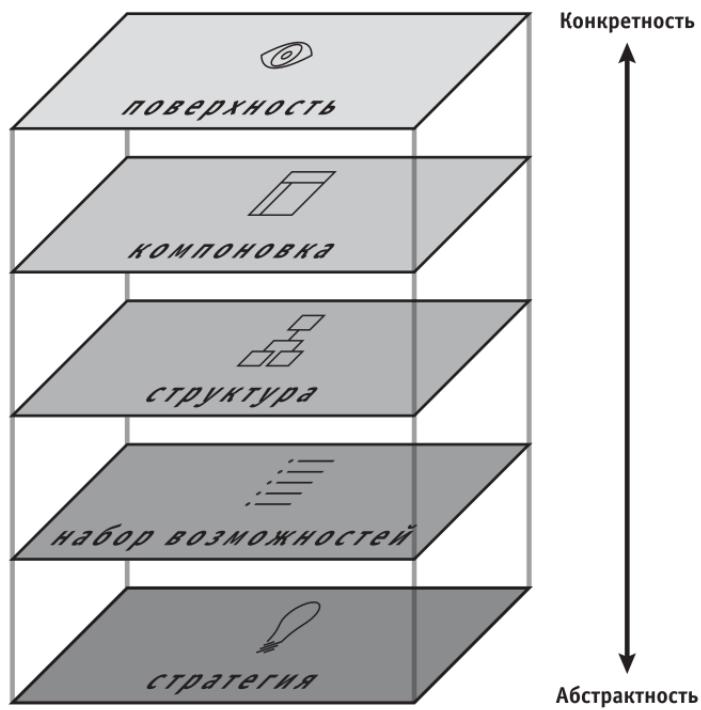
Пять уровней UX – это концептуальная модель, предложенная Джессом Гарреттом (Jesse James Garrett) для **проектирования опыта пользователя** веб-приложений [9]. Однако её можно расширить и на проектирование продукта вообще.

Процесс разработки и планирования UX разбивается на несколько этапов, каждый из которых рассматривает UX на разном уровне абстракции.

- Уровень поверхности (surface)
- Уровень компоновки (skeleton)
- Уровень структуры (structure)
- Уровень возможностей (scope)
- Уровень стратегии (strategy)

Самый первый и абстрактный уровень – понимание пользователей и их нужд. Продукт с хорошим интерфейсом, который не решает никаких задач пользователей – бесполезен. Здесь же

Рисунок 4.1. Пять уровней проектирования UX



мы должны понять цели заказчика, если он и пользователи не одно и то же лицо. Цель пользователя интернет магазина – купить нужный товар. Цель владельца – продавать товары. Цели пользователей и заказчика могут быть одинаковыми или немного различаться. Владельцы интернет-магазина хотят продать больше товара и привлечь больше покупателей, но не все покупатели согласны потратить лишние деньги на покупки или получать рекламную рассылку.

Когда понятно, чего хотят пользователи, нужно продумать функциональность продукта и информацию, которую он предоставит пользователю. Всё это должно позволить пользователям достичь своих целей – это уровень возможностей.

Когда готовы требования к функционалу и информационному наполнению нужно понять, на какие окна экраны или страницы разделить программу или сайт. То есть описать информационную архитектуру. Это особенно актуально для сайтов, где обычно много страниц.

Определившись с содержимым каждой страницы или окна, определяют компоновку блоков из элементов интерфейса. Создают грубые макеты в которых отмечено положение основных групп элементов ПИ. Макетов обычно делают несколько вариантов. Их основная задача – понять, какой вариант компоновки больше подходит для решения задач пользователями.

Наконец выбирают наиболее удачные варианты макетов и дополняют их деталями, приводя уже конечный вид интерфейса пользователя, добавляя имитацию всех или отдельных действий – получая прототип.

Полученные макеты или прототипы, совместно с необходимыми материалами – изображениями, набором иконок и описанием стилей или дизайн-системой передают непосредственно разработчикам (например, верстальщикам), которые будут реализовывать готовый продукт.

Разбивка проектирования UX, а с ним и продукта на эти этапы позволяет ограничиться принятием в самом начале более стратегических решений, не обращая внимания на детали реализации. Далее, развивая каждое из принятых решений, проводят продукт к его конечному виду (рис. 4.2).

Такой подход не минимизирует риск появления ошибок планирования. Например, если страницы сайта проектируются или даже верстаются на этапе уточнения требований заказчика или исследования потенциальных пользователей (уровень стратегии), вполне вероятно, что часть созданных страниц окажутся лишними, а часть придётся переделать. Конечно, если продукт, например, сайт интернет-магазина, типовой, риск таких ошибок небольшой. Но только потому, что все типовые решения прошли все стадии проектирования в том или ином виде. Кроме того, есть риск, что заказчику и конечным пользователям типовое решение в чистом виде не подойдёт.

Опишем каждый из этапов разработки более подробно.

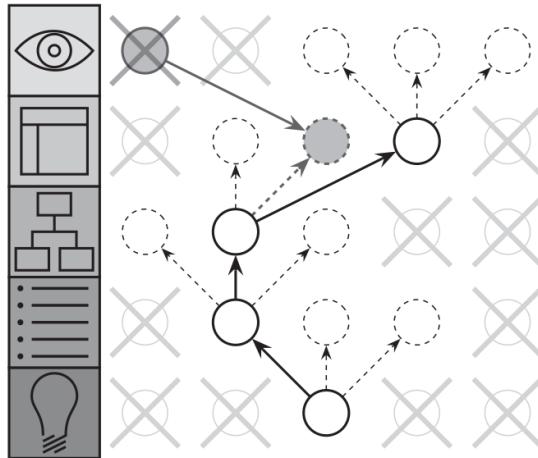


Рисунок 4.2. Принимая решения на каждом уровне проектирования, разработчик сужает диапазон решений на каждом следующем уровне. Это упрощает проектирование. Решения, принятые на разных уровнях одновременно, могут быть не согласованы.

4.2.2 Уровень стратегии

Первый этап проектирования продукта – определиться с целями и задачами пользователей, в том числе потенциальными и понять цели заказчика (если он есть).

Здесь возможны несколько вариантов:

1. заказчик и пользователи – одно лицо. Пример: разработка продукта для использования внутри компании заказчика. Например, ПО для ERP³ системы.
2. Заказчик отдельно, пользователи отдельно. Пример – разработка интернет магазина.
3. Заказчиком выступает сам разработчик.

Изучение заказчика. Изучение целей заказчика – бренд, преследуемые цели. Посмотрите брендбук⁴. Возможно, понадобится написать для заказчика техническое задание⁵.

Заказчик не всегда стремится удовлетворить потребности поль-

³Enterprise Resource Planning, планирование ресурсов предприятия. 1С: Предприятие – пример ERP ПО

⁴Брендбук – официальный документ компании, в котором описывается концепция бренда, и другие данные, включая полное руководство по фирменному стилю

⁵Техническое задание (ТЗ, техзадание) – документ или несколько документов, определяющих цель, структуру, свойства и методы какого-либо проекта, и исключающие двусмысленное толкование различными исполнителями

зователя. Иногда он стремится эти потребности создать. Например, предлагает дополнительные товары в интернет-магазине, делает таргетированную рекламу.

Одна из проблем дизайна продуктов – дизайн продуктов должен быть предназначен для пользователей, но чтобы быть реализованным он должен понравиться заказчику. Который, как правило, неспециалист в области проектирования интерфейсов.

Изучение целевой аудитории. В первую очередь нужно узнать о целях пользователей и задачах, которые они решают. Как правило, это можно узнать у заказчика, если он хочет использовать продукт сам или хорошо представляет целевую аудиторию. Если продукт узкоспециализированный то имеет смысл провести опрос или проинтервьюировать пользователей.

Пользователи могут предлагать способы решения своих задач. Но стоит помнить, что пользователи – не эксперты в области проектирования интерфейсов, с другой стороны, разработчик, скорее всего, тоже не эксперт в предметной области пользователей.

На этом этапе разумно изучить аналогичные продукты.

В конце этого этапа нужно ответить на главные вопросы – для чего и для кого разрабатывается продукт.

4.2.3 Метод персонажей

Часто целевая аудитория продукта неоднородна, поэтому усреднённый пользователь будет слишком нереалистичной моделью. Поэтому целевую аудиторию разделяют на несколько групп по принципу схожести целей и опыта пользователей. Для каждой группы описывается типичный представитель – **персонаж** (персона). В зависимости от целевой аудитории составляют 2-

6 персонажей.

Метод персонажей был предложен Аланом Купером. Он помогает структурировать целевую аудиторию продукта, задокументировать потребности и особенности значимых групп пользователей. Предполагается, что продукт проектируется для персонажей. Этот подход также используется в маркетинге.

Персонажи описываются на основе исследования потенциальных пользователей. Если данных о целевой аудитории недостаточно, то условный персонаж, описание которого частично состоит из гипотез, тоже чаще всего оказывается полезен.

Описание персонажа обычно включает в себя:

- Имя;
- Фотографию (аватар);
- Демографические данные: пол, возраст, образование, род деятельности, семейный статус, дети;
- Пользовательский опыт (опытный пользователь или новичок?), используемые устройства
- Цели – зачем пользователи будут использовать продукт?
- Мотивация – почему пользователь будет использовать именно ваш продукт?
- Барьеры – что мешает пользователю достигать свои цели?

Конкретный набор характеристик персонажа может меняться в зависимости от продукта. Проектируя интернет-магазин подарков семейные статус и наличие детей будут иметь значение, а для сайта тематического интернет-издания нет.

Помимо перечисленных характеристик персонажа иногда имеет смысл добавить описание контекста, в котором пользователь будет использовать продукт. ГИС часто используют на ходу, со смартфона, значит важно понимать, что интерфейс дол-

жен быть адаптирован для использования одной рукой, все детали должны быть различимы при ярком солнечном свете. Стоит указать какие аккаунты имеет персонаж, в каких социальных сетях персонаж зарегистрирован. Для тематического новостного сайта тогда можно будет предусмотреть репост в социальную сеть и возможность регистрации на сайт через Гугл-аккаунт. Какими ещё продуктами пользуется персонаж? Если разрабатывается CRM-система, то возможно стоит создать интерфейс, знакомый пользователям продуктов 1С. К целям персонажа как пользователя иногда имеет смысл добавить цели персонажа как личности.

Карточка персонажа может быть оформлена в текстовом редакторе, представлена в виде ментальной карты или создана в специальном сервисе, например uxpressia.com, hubspot.com/make-my-persona, xtensio.com.

Таким образом, персонажи помогают структурировать информацию о целевой аудитории и понять её потребности. Дают возможность оценивать проектные решения с точки зрения архетипических пользователей и удерживают от соблазна проектировать для себя, но не для целевой аудитории. Наконец, способствуют эмпатии UX-дизайнера по отношению к пользователю.

Пример персонажа – пользователя сайта вуза.

Дмитрий, 25 лет

Работает системным администратором, студент, учится заочно.

Цели:

- Хочет получить образование и диплом бакалавра в ИТ (программирование).
- Хочет знать даты сессии, расписание занятий и экзаменов.
- Хочет получать задания дистанционно.

- Хочет знать, правильно ли он понял задание
- Хочет знать как оформить контрольную работу, курсовую работу.
- Хочет знать, выбрал ли он правильный способ выполнения заданий

Барьеры, демотивирующие факторы:

- Не любит узнавать неожиданные новости в последний момент, перед самой сессией (смена дат экзаменов, изменение заданий)
- Не запоминает, где находится нужная информация на сайте университета
- Не хочет делать задания по "ненужным" предметам.
- Иногда откладывает дела на последний момент.

Пользовательский опыт. Опытный пользователь, активно пользуется социальными сетями.

Используемые устройства: смартфон с ОС Андроид 10, ноутбук с ОС Windows 8.

Диаграмма прецедентов

Для описания функциональных возможностей может использоваться **use case** диаграмма (**диаграмма прецедентов**)

Диаграмма вариантов использования (use case diagram, диаграмма прецедентов) – UML-диаграмма, отражающая отношения между актёрами и прецедентами [13].

Она строится из трёх основных элементов:

- **Система** (рамки системы). Прямоугольник обозначающий систему. Включает в себе прецеденты. Иногда не приводится.

- **Прецедент** – возможность системы (часть её функциональности), благодаря которой пользователь может получить конкретный, измеримый и нужный ему результат.
Обозначается овалом с описанием возможности из 1-3 слов.
Например: просмотр страниц сайта, комментирование, добавление постов.
- **Роль** (actor), которую пользователь играет по отношению к системе. Обозначается человечком с названием роли. Например: посетитель сайта, администратор сайта

Роль – модель пользователя, которая концентрируется на наборе возможностей, а не на характеристиках самих пользователей (персонажей). Потребности каждого персонажа должны быть реализованы через прецеденты. При этом одной роли могут соответствовать несколько персонажей, так же как и персонаж может менять роли.

Прецедент соответствуетциальному сервису системы, определяет один из вариантов её использования и описывает типичный способ взаимодействия пользователя с системой.

Для примера приведём фрагмент диаграммы вариантов использования (рис. 4.3) банковского приложения, которая иллюстрирует основные элементы и отношения между ними, но не претендует на полноту. У пользователя могут быть две роли: клиент, VIP клиент. Стрелка к первой роли – отношение обобщения – означает, что VIP клиент имеет те же возможности (дополняя их своими), что и обычный клиент.

Выполнение операций пополнения счёта и снятия средств обязательно включает (стрелка include к новому прецеденту) в себя отображение нового баланса. Нельзя изменить баланс без показа нового баланса. Обновление процентной ставки может расширить (extend) прецедент "Пополнение счёта". Но это происходит только при соблюдении условия (пунктирный прямоугольник). Можно пополнить счёт и при этом не изменять про-

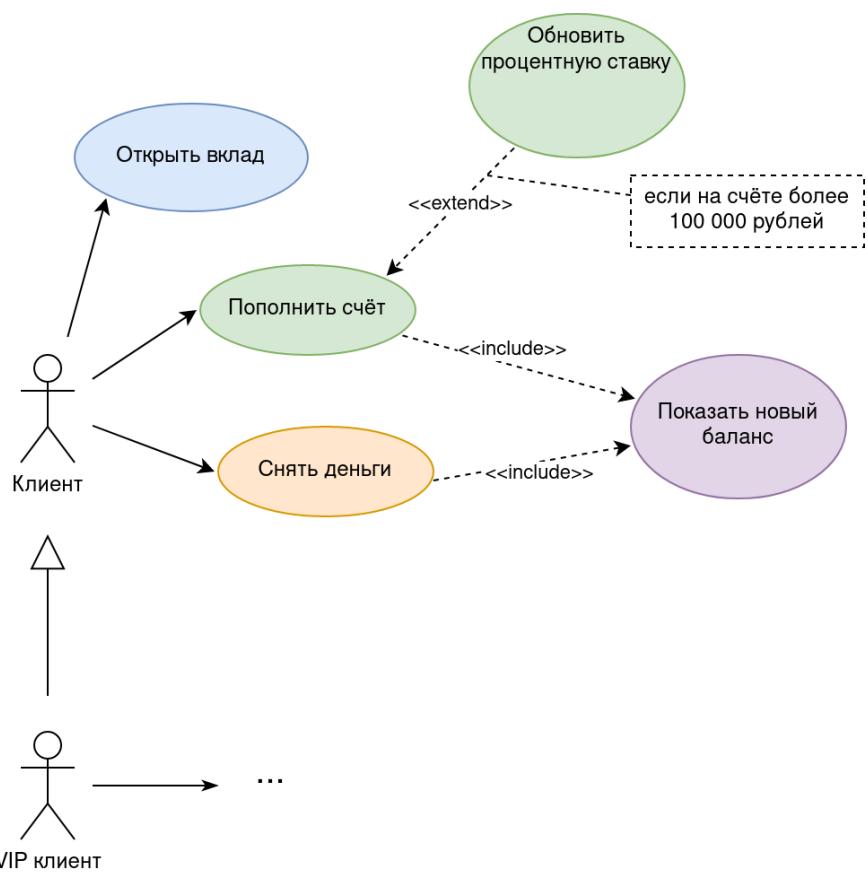


Рисунок 4.3. Фрагмент диаграммы вариантов использования для банковского приложения

центную ставку. Подобные отношения включения могут быть и безусловным: клиент может заказать документ с отчётом после изменения баланса. Отдельные группы прецедентов можно раскрасить, обозначив их разный смысл.

Диаграмма вариантов использования ничего не говорит об устройстве интерфейса. Но нужно понимать, какие роли и возможности будут у пользователей, чтобы проектировать интерфейс.

4.2.4 Уровень структуры

На этом уровне описывается информационная архитектура продукта в представлении пользователя, основные этапы (экраны, окна или страницы в зависимости от типа продукта) взаимодействия пользователя и программы. Далее будут рассмотрен самый популярный тип интерфейса пользователя – графический интерфейс. Создание текстового, голосового, или командного (частный случай текстового) пользовательских интерфейсов не вполне укладывается в стратегию, предложенную Гарреттом.

Информационная архитектура (Information architecture) – сочетание схем организации, предметизации и навигации, реализованных в информационной системе.

Во время создания продукта обычно описывается информационная архитектура в нескольких представлениях: диаграмма классов, модулей, структура БД и другие. Это модели реализации. Проектируя пользовательский интерфейс, нужно описать модель представления – то, как продукт выглядит с точки зрения пользователя.

Информационная архитектура описывается диаграммой экранов, страниц или окон. Она состоит из блоков, обозначающих экран, с названием и кратким описанием их содержимого и переходов между этими экранами. Такая диаграмма похожа на

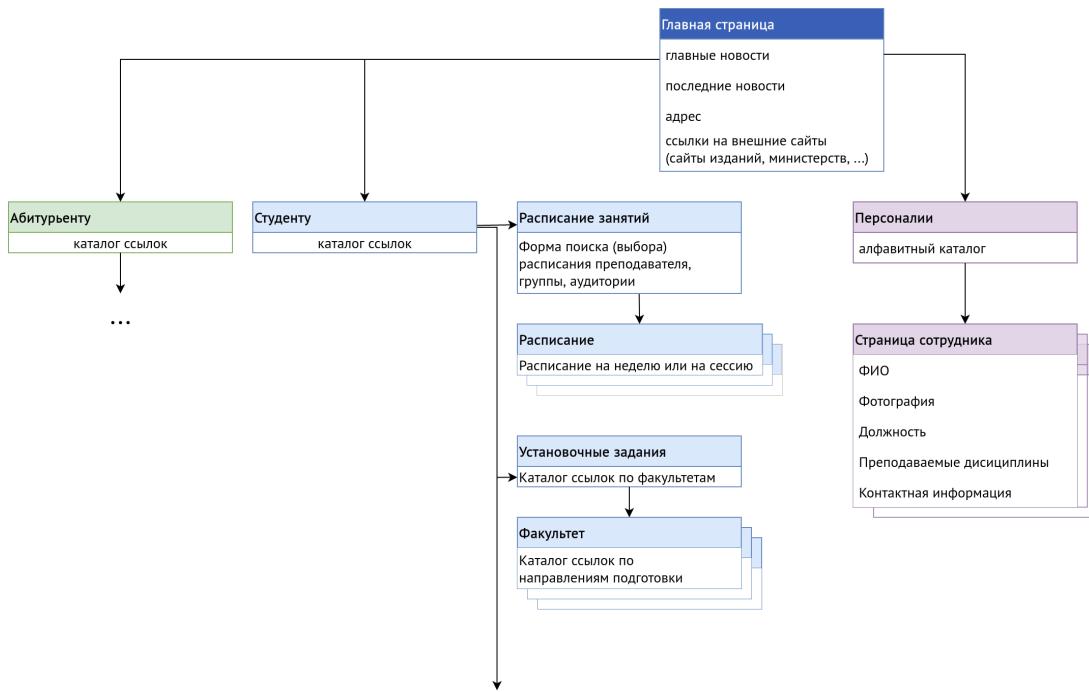


Рисунок 4.4. Фрагмент диаграммы страниц сайта университета. "Страница сотрудника" "Факультет" "Расписание"abor страниц с одинаковым назначением, но разным содержанием.

карту сайта.

По диаграмме экранов можно проследить выполнение всех прецедентов из диаграммы вариантов использования, оценить на сколько удобна и понятна такая структура для решения типичных задач персонажей.

Выполнение сценариев использования (todo: добавить определение и описание; на каком этапе они описываются?) и системы на уровне структуры описывается в виде путей пользователя (user journey) и пользовательских сценариев (user flow).

4.2.5 Уровень компоновки

На этом этапе определяется общий вид интерфейса интерфейс пользователя, примерное расположение основных элементов интерфейса пользователя. Компонуются экраны после составления диаграммы информационной архитектуры. Каждый экран представляется низко детализированным макетом, главная цель которого – показывать взаимное расположение элементов, но

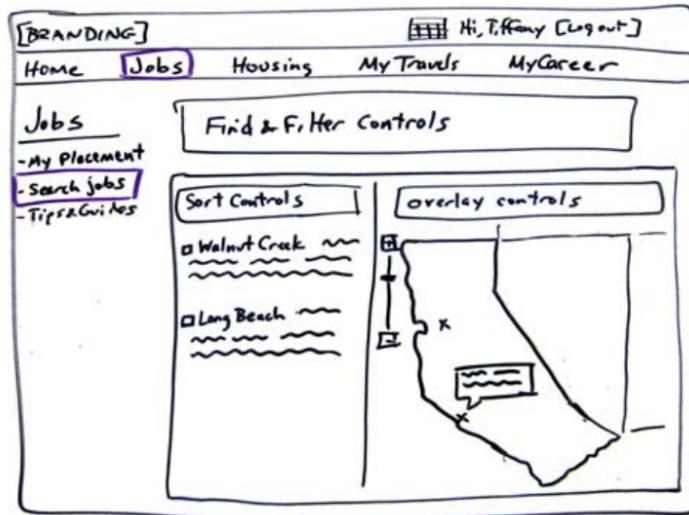


Рисунок 4.5. Эскиз окна приложения: видна компоновка элементов интерфейса, основные элементы подписаны, изображение приведено схематично, палитра не задана

не их конечный вид (рис. 4.5).

Низко детализированные эскизы полезны для быстрого со-здания экранов. Легко создавать несколько вариантов одного экрана (рис. 4.6), менять их дизайн, не подгоняя элементы друг к другу. Дизайнер экономит время, откладывая выбор палитры, шрифтов, точных надписей и текста, иконок и т.д. для всех вариантов макетов.

На этом этапе полезно утверждать макеты у заказчика (если есть такая возможность) потому, что при более детальном проектировании интерфейса цена изменений возрастает.

Эскизы удобнее всего создавать в специальном ПО для дизайна интерфейсов. Но можно рисовать макеты или эскизы от руки или в графическом редакторе.

Изменять макеты или создавать много вариантов удобнее всего именно в специальном ПО. Такие программы имеют встроенные инструменты для группировки и точного задания по-ложения, повторного использования элементов интерфейса и средства для быстрой смены их стилей. SketchUp, Figma – са-мые популярные программы на сегодняшний день [57].

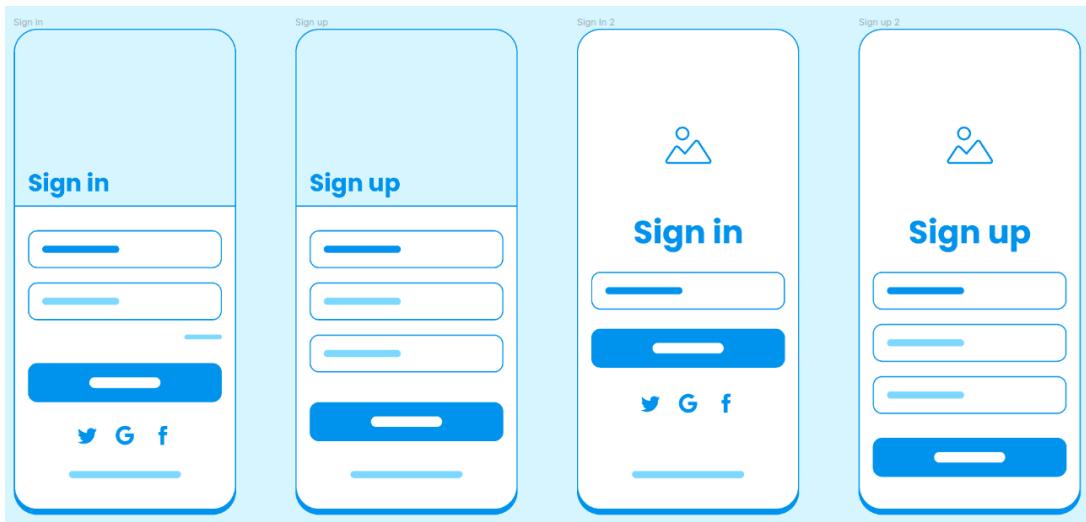


Рисунок 4.6. Низко детализированные варианты экранов приложения. Выполнены в Figma, собраны из библиотеки готовых компонентов. Обозначены элементы интерфейса, расположение отдельных надписей, выбранная палитра играет техническую роль и не отражает конечный вид интерфейса.

Макеты можно нарисовать и от руки, затем легко вырезать из бумаги отдельные элементы интерфейса и экспериментировать с разными компоновками. Такое исполнение макета интерфейса для мобильных устройств с тачскринами даёт представление о физическом взаимодействии с интерфейсом (рис. 4.7).

4.2.6 Уровень поверхности

На этом этапе интерфейс получает свой окончательный вид. Должна быть выбрана цветовая палитра, нарисованы иконки и созданы изображения, окна или страницы наполнены содержанием.

Всё вышеописанное представляется в окончательном макете (рис. 4.8 или прототипе). Прототип, в отличие от макета, предполагает имитацию взаимодействия с пользователем. Например, интерактивные переходы со страницы на страницу, прокрутку внутри окон, раскрытие меню и т.п.

Готовый макет используется для реализации интерфейса программы или сайта.

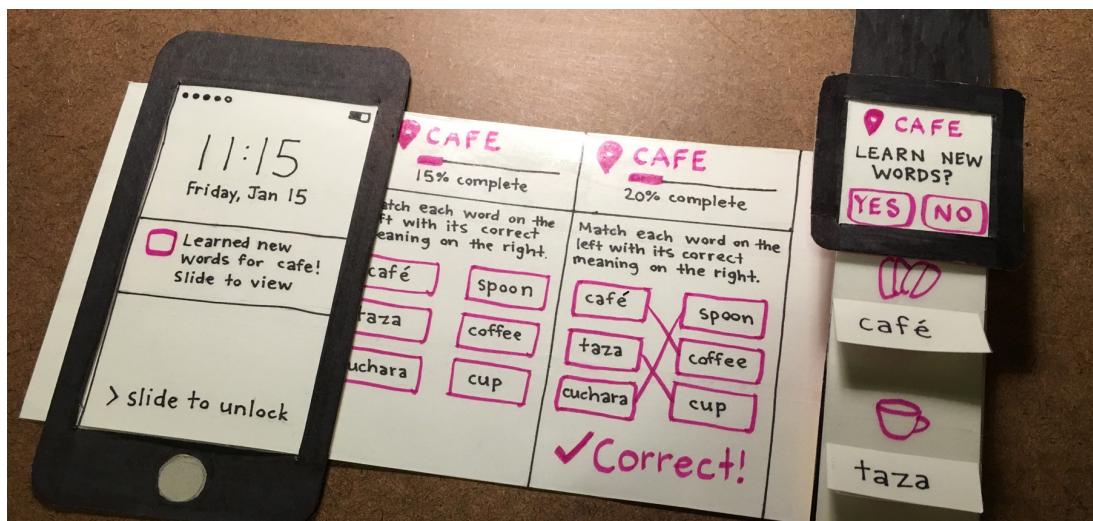


Рисунок 4.7. Бумажный эскиз пользовательского интерфейса

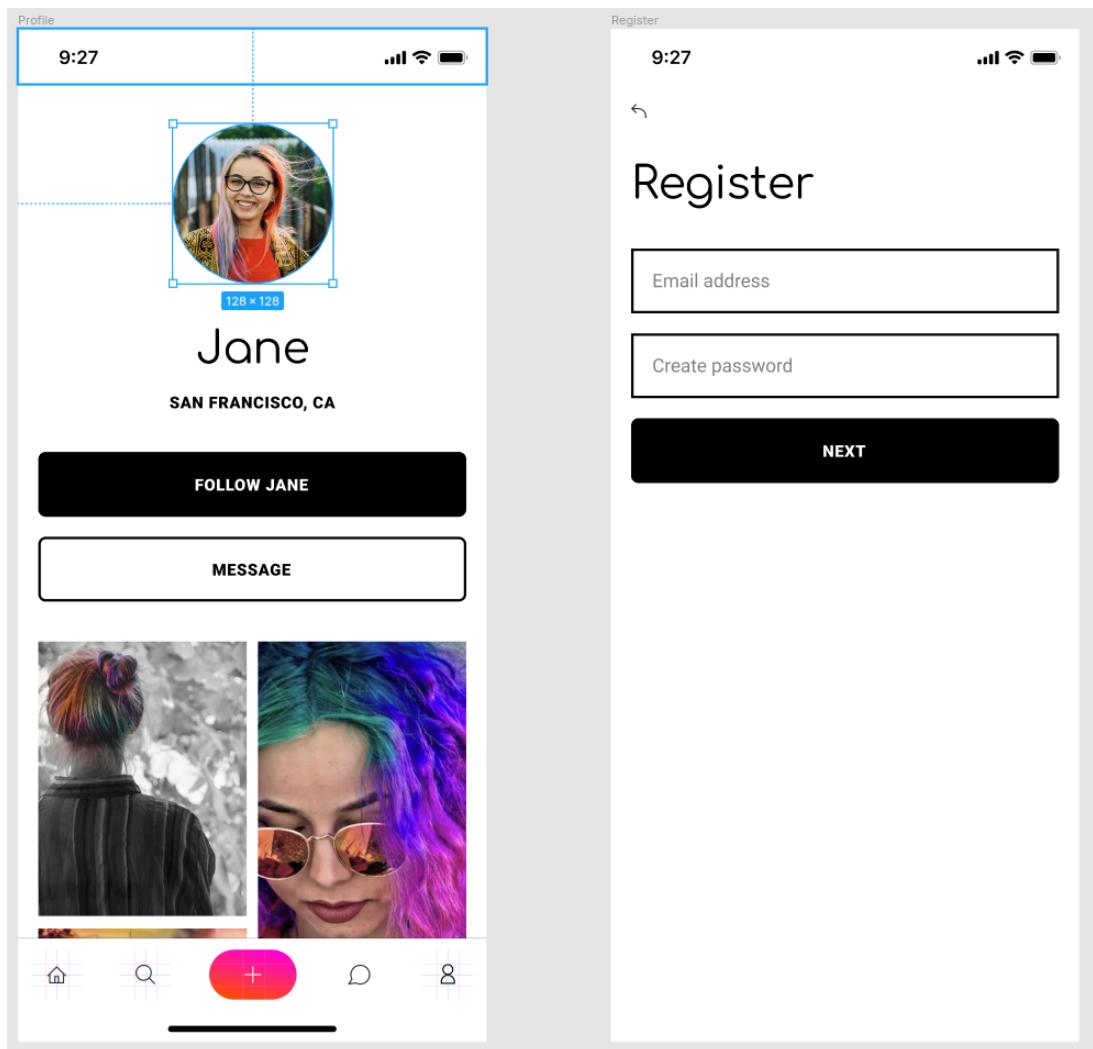


Рисунок 4.8. Макет пользовательского интерфейса в Figma — онлайн-сервисе для разработки интерфейсов

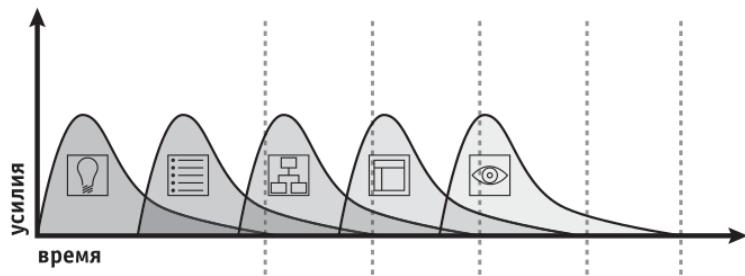


Рисунок 4.9. Каждый следующий этап проектирования должен начинаться до полного завершения предыдущего.

4.3 Методология проектирования

Программы или сайты, решающие довольно простые задачи, можно создать пропуская многие этапы проектирования. Время, необходимое на исправление или изменение интерфейса таких продуктов может быть сильно меньше, чем время затраченное на проектирование интерфейса. Но для относительно крупных продуктов своевременное проектирование (определение перечня решаемых задач, информационной архитектуры и пользовательского интерфейса) позволяет избежать ошибок в дальнейшем.

Проектирование сверху вниз не обязательно должно иметь чётко очерченные этапы, где каждый следующий этап зависит только от предыдущих. Возможна корректировка решений, принятых на предыдущих этапах. Поэтому стоит начинать работать над новым этапом проектирования, когда предыдущий ещё полностью не завершился (рис. 4.9).

Например, создавая макет страницы (уровень компоновки) с описанием направлений подготовки (специальностей) вуза, разработчик может выяснить, что направлений слишком много. Пользователям будет проще ориентироваться если создать отдельные страницы факультетов или кафедр. Значит, нужно изменить диаграмму, описывающую информационную архитектуру.

Методология проектирования предложенная Джесси Гаретом

– не единственно верная. Но почти у всех методологий проектирование общая идея – проектирование идёт сверху вниз, от проектирования наиболее абстрактных аспектов (например, набора прецедентов) к наиболее конкретным (отдельным элементам пользовательского интерфейса). Более детальная схема (рис. 4.10) всех этапов проектирования дана, например, Аланом Купером в книге Основы проектирования взаимодействия [10].

Вопросы

1. Что такое UX? Что входит в это понятие?
2. Назовите этапы проектирования (по Джессу Гаретту), опишите каждый этап.
3. Зачем нужно моделировать потенциальных пользователей?
4. Из чего должно состоять описание персонажа?
5. На основе какой информации описывается персонаж?
6. Что такое диаграмма прецедентов? Зачем она нужна?
7. Какие бывают модели пользователя?
8. Из каких элементов строится диаграмма прецедентов?
9. Опишите виды отношений между прецедентами и ролями.
10. Как описать информационную архитектуру?
11. Зачем создавать низко-детализированные макеты пользовательского интерфейса?
12. Какие средства существуют для создания макетов?
13. Как должно распределяться время между этапами проектирования?

Литература

- Алан Купер, Рейманн Роберт М., Основы проектирования взаимодействия. – Пер. с англ. – СПб.: Символ-Плюс, 2018,

720 с.

- Гарретт Дж. Веб-дизайн: Элементы опыта взаимодействия». – Пер. с англ. – СПб.: Символ-Плюс, 2008. – 192



Целеориентированное проектирование				
	Активности	Объекты внимания	Участие лиц, принимающих решения	Сдаваемый результат
Исследования	Охват Определение целей и графика проекта	Задачи, сроки, финансовые ограничения, процесс, контрольные точки	Совещания Оценка технических возможностей и затрат	Документ Отчет о проделанной работе
	Аудит Изучение доступных документов и существующих продуктов	Бизнес-планы и маркетинговые планы, стратегия брендинга, маркетинговые исследования, планы развития продуктовой линейки, конкуренты, технологии в соответствующей области		
	Интервью с лицами, принимающими решения Прояснение образа продукта и имеющихся ограничений	Образ продукта, риски, благоприятные возможности, ограничения, логистика, пользователи	Интервью С лицами, принимающими решения, и пользователями	
	Интервью и наблюдения за пользователями Прояснение потребностей пользователей и изучение их поведения	Пользователи, потенциальные пользователи, модели поведения, взгляды, способности, мотивы, характеристики окружения, инструменты, проблемы	Приемка Предварительных результатов исследований	
Моделирование	Персонажи Создание архетипов пользователей и клиентов	Шаблоны поведения пользователей и клиентов, взгляды, способности, цели, характеристики окружения, инструменты, проблемы	Приемка Персонажей	
	Прочие модели Представление особенностей предметной области, не связанных с отдельными пользователями и клиентами	Рабочие процессы, затрагивающие группы людей, характеристики окружения, артефакты		
Выработка требований	Контекстные сценарии Сочинение историй об идеальном опыте пользователей	Как продукт соответствует жизни и среде персонажей и помогает им в достижении целей	Приемка Сценарiev и требований	
	Требования Определение критичных возможностей продукта	Функциональные и информационные потребности, ментальные модели пользователей, императивы проектирования, образ продукта, бизнес-требования, технология	Презентация Анализ пользователей и предметной области	Документ Анализ пользователей и предметной области
Проектирование инфраструктуры	Элементы Описание информационной и функциональной модели	Информация, функции, механизмы, действия, объектные модели предметной области	Приемка Общей структуры проекта	
	Инфраструктура Проектирование общей структуры опыта взаимодействия	Связи между объектами, концептуальные группы, навигационные последовательности, принципы и шаблоны, поток управления, эскизы, раскадровки		
	Ключевые и проверочные сценарии Описание взаимодействия персонажа с продуктом	Как дизайн продукта соответствует идеальной последовательности действий пользователя и учитывает разнообразие вероятных условий	Презентация Концепция пользовательского интерфейса	
Детализация	Детальное проектирование Доработка и уточнение деталей	Внешний вид, идиомы, интерфейс, виджеты, поведение, информация, визуализация, бренд, опыт, язык, раскадровки	Приемка Детального проекта	Документ Спецификация формы и поведения
Сопровождение проектирования	Корректировка спецификации Учет новых ограничений и сроков	Поддержание концептуальной целостности дизайна продукта в условиях меняющихся технологических ограничений	Совместное проектирование	Пересмотренный документ Спецификация формы и поведения

Рисунок 4.10. Этапы проектирования продукта

5 Юзабилити и тестирование

5.1 Юзабилити

Юзабилити (usability, удобство использования, эргономичность) — способность продукта быть понимаемым, изучаемым, используемым и привлекательным для пользователя в заданных условиях.

Рассмотрим 10 правил (эвристик¹) Якоба Нильсона, помогающих добиться хорошего юзабилити.

1. Видимость статуса системы. Пользователь должен всегда знать, что происходит, получая подходящую обратную связь в приемлемое время.

Видимость статуса системы: Выполнение операции

- Если отклик программы более 200 мс покажите индикатор загрузки, смените форму курсора.
- Если программа выполняет действие дольше нескольких секунд, то покажите полосу прогресса.
- Если в программе предусмотрены режимы, то явно обозначьте текущий режим.
- Пример режимов: режим вставки или перезаписи в текстовом редакторе (вертикальный или горизонтальный курсор); режим кисти и резинки в графическом редакторе.

2. Соответствие между системой и реальным миром. Си-

¹ Эвристика – формально не обоснованное, но работающее на практике правило

Константинов

Имя (обязательное поле)

Константин

Отчество (обязательное поле)

Константинович

Не имею отчества

Пол (обязательное поле)

Женский

Мужской

Дата рождения (обязательное поле)

09 / 02 / 1980

Внимание

В анкете должны быть заполнены все обязательные поля

Понятно

Рисунок 5.1. Плохая видимость статуса системы. Анкета на сайте leadersofdigital.ru: обратная связь есть, но не полная. Непонятно какие поля не заполнены. Решение: вместо диалогового окна с сообщением, выделить (например, цветом) незаполненные поля, возможно добавить к каждому полю комментарий поясняющий формат (например: введите дату в формате: ДД / ММ / ГГГГ)

* = Required

* First Name:	<input type="text" value="Chuck"/>	Street Address Is Required
* Last Name:	<input type="text" value="Woolry"/>	
Business Name:	<input type="text"/>	
*Street Address: (no P.O. Boxes)	<input type="text"/>	
Suite/Apt:	<input type="text"/>	
*City:	<input type="text" value="Hollywood"/>	
*State:	<input type="text" value="CA"/> CALIFORNIA	
*Zip Code:	<input type="text" value="90210"/>	
*Phone number:	<input type="text" value="(310) 123-3234"/>	

APO may incur additional shipping charges.

Рисунок 5.2. Название поля выделено. В подсказке справа, при необходимости, можно уточнить, что именно не так ввёл пользователь. На такие подсказки не нужно лишний раз кликать, в отличие от диалоговых окон. Они точно указывают на ошибку.

4129 Chestnut Street

CITY	STATE	ZIP
San Francisco	California	9

CONTACT INFO

BILLING PHONE NUMBER ?

If we have order questions
Oops. Looks like you forgot your Billing Phone Number.

Рисунок 5.3. Незаполненное поле выделено. Подсказка дана снизу. Нужно немного увеличить размер текста подсказки.

Заполните форму и мы свяжемся с Вами

Ваше имя*

Ошибка! Поля обязательные для заполнения.

Адрес эл.почты*

Ошибка! Поля обязательные для заполнения.

Номер телефона

Введите ваше сообщение

Ошибка! Поля обязательные для заполнения.

* - обязательное поле



Рисунок 5.4. Плюсы: отмечены все незаполненные поля. Минусы: форма "кричит" на пользователя: Ошибка! Ошибка! Ошибка! Решение: тактично сообщить об ошибке, например "Пожалуйста заполните поле"; выбрать менее насыщенный цвет для фона всех полей.

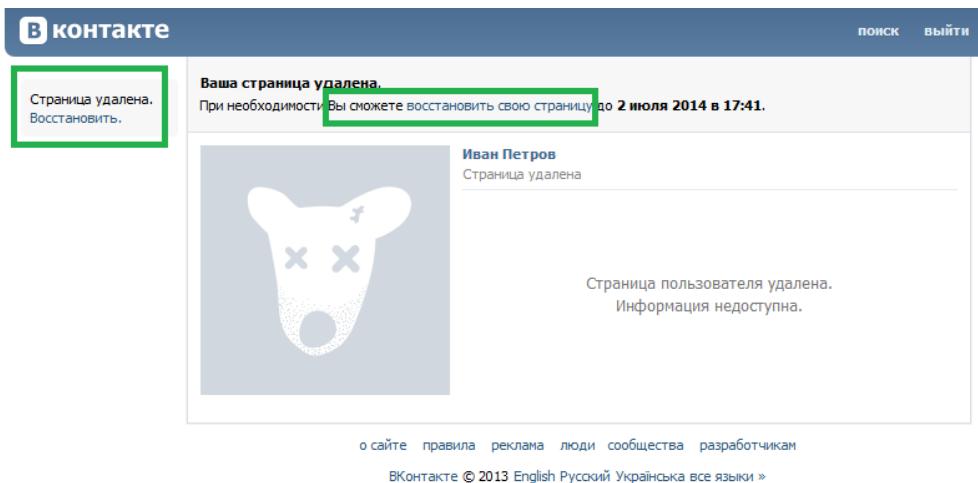


Рисунок 5.5. vk.com откладывает удаление данных на некоторое время, чтобы у пользователя была возможность отменить действие

система должна «говорить на языке пользователя», используя понятную ему терминологию и концепции.

- Не используйте специальные термины из ИТ: авторизация (вход), транзакция (операция, запрос), лейбл (надпись), ...
- Страйтесь не использовать специальные термины, из предметной области без необходимости. Например, в мобильном приложении банка для обычных пользователей: дебиторы, кредиторы, контрагенты, ...

3. Управляемость и свобода для пользователя

Пользователи неизбежно ошибаются. Нужно дать понятную возможность исправить ошибку (рис. 5.5), например, функции отмены (undo) и повтора (redo).

4. Согласованность и стандарты Пользователи не должны гадать, значат ли одно и то же разные слова, ситуации или операции. Также нужно следовать соглашениям, принятым для данной платформы.

- Используйте одни и те же слова и обозначения для одних и тех же действий.
- Следуйте дизайн-системе²

²Дизайн-система — набор компонентов, правил, предписаний по дизайну

- Сделайте интерфейс предсказуемым.

5. Предотвращение ошибок

Продуманный дизайн, который не позволяет какой-то проблеме даже возникнуть, лучше, чем самые хорошие сообщения об ошибках. Следует устранять сами условия возникновения ошибок, либо выявлять их и предупреждать пользователя о предстоящей проблеме.

6. Распознавать лучше, чем вспоминать

Минимизируйте нагрузку на память пользователя, явно показывая ему объекты, действия и варианты выбора. Пользователь не должен в одной части диалога запоминать информацию, которая потребуется ему в другой. Инструкции по использованию системы должны быть видимы или легко доступны везде, где возможно.

7. Гибкость и эффективность использования

Программа должна быть понятной для новичков и эффективной для профессионалов. Акселераторы (средства быстрого выполнения команд), которые новичок даже не видит, для опытного пользователя часто могут ускорить взаимодействие. Поэтому система должна удовлетворять как неопытных, так и опытных пользователей. Следует давать возможность настраивать под себя часто используемые операции.

8. Эстетический и минималистический дизайн

В интерфейсе не должно быть информации, которая не нужна пользователю или которая может понадобиться ему в редких случаях. Каждый избыточный элемент диалога отнимает внимание от нужных элементов, затрудняет визуальный поиск.

9. Помочь пользователю понять и исправить ошибку

Сообщения об ошибках следует писать простым языком, без кодов, чётко формулируя проблему и предлагая конструктивное решение.

10. Справка и документация

Хотя было бы лучше, если бы система была пригодна к использованию без документации, всё же необходимо предоставлять справку и документацию. Информация должна быть простой в поиске, соответствовать задаче пользователя, описывать конкретную последовательность действий, и не должна быть слишком большой.

5.2 Юзабилити-тестирование

5.2.1 Понятие юзабилити-тестирования

Юзабилити-тестирование, (usability testing) — исследование, выполняемое с целью определения, удобен ли некоторый искусственный объект (такой, как веб-страница, пользовательский интерфейс или устройство) для его предполагаемого применения.

Это тестирование продукта, но не исследование целевой аудитории и не тестирование пользователей. Так как продукт к этому времени должен быть завершён хотя бы частично, а значит целевая аудитория уже должна быть изучена, известны её потребности. В результате проведённого тестирования дорабатывается пользовательский интерфейс продукта, но не изменяется целевая аудитория.

Юзабилити-тестирование должно проводится при участии пользователей, а не вовлечённых в разработку людей. У пользователей, в отличии от разработчиков, другой набор ментальных моделей, они не осведомлены о внутреннем устройстве продукта.

Провести тестирование можно большим набором способов. Начиная от неформального общения с непосредственными пользователями, до строгих методов юзабилити-тестирования с наблюдением за пользователем, решающим конкретную задачу. В последнем случае действия пользователя могут быть записа-

ны, а потом проанализированы с составлением журнала действий и измерением времени этих действий.

Неформальные методы юзабилити-тестирования проще организовывать, можно проводить относительно часто. С другой стороны, разработчик продукта обычно осознанно или нет вмешивается в действия пользователя, что искажает результаты.

5.2.2 А/В-тестирование

Идея А/В-тестирования³ заключается в следующем:

- пользователей разделяют на две (или более) группы
- каждой из групп показывай свой вариант (А или В) интерфейса
- варианты интерфейса различаются чаще всего в деталях, например расположением отдельных элементов, цветами и .т.п.
- пользователи часто не подозревают об участии в тестировании и о существовании разных вариантов интерфейса
- после сбора достаточной большой статистики сравниваются показатели (например, конверсия⁴, время проведённое на сайте и др.)

5.2.3 Тестирование с наблюдением

Перед тестированием составляется список задач, для которых создавался продукт и которые предлагаются пользователям для решения. Процесс тестирования фиксируется в протоколе тестирования. Где отмечается трудности, с которыми столкнулся пользователь: непонимание инструкций, ответов системы и

³Термин изначально появился в маркетинге

⁴Конверсия — это отношение (в процентах) числа посетителей сайта, выполнивших на нём какие-либо целевые действия, к общему числу посетителей сайта



Рисунок 5.6. А/В тестирование: в течение некоторого времени для разных пользователей веб-страница показывалась в одном из двух вариантов; результат эксперимента: пользователи, которым показывали второй вариант страницы, чаще делали заказ.

Рисунок 5.7. Два варианта формы регистрации на сайте букмекерской компании. На форме справа добавлено: Мы даём 100% гарантию безопасности ваших данных. Они никому не будут переданы. Результат эксперимента: увеличение количества регистраций на 20% по сравнению с оригинальной формой. Объём выборки: 20257 пользователей, 380 регистраций.

т.д. Протокол тестирования может включать видеозапись экрана.

Наблюдение за тем, как люди взаимодействуют с продуктом, нередко позволяет найти для него более оптимальные решения. Если при тестировании используется модератор, то его задача — держать респондента сфокусированным на задачах (но при этом не «помогать» ему решать эти задачи).

Вариация метода – оценка мест, где пользователь при решении задачи отклонился от идеального сценария, описанного разработчиками продукта.

В качестве пользователей обычно выбираются люди, соответствующие разработанным ранее пользовательским моделям. Например, персонажам.

На практике далеко не всегда есть возможность пригласить потенциальных пользователей для тестирования. Тем более, если это не сотрудники заказчика, для которых продукт и разрабатывается, и которые могут быть заинтересованы в тестировании. В этом случае прибегает к так называемому коридорному тестированию. Когда к тестированию привлекается первый попавшийся человек “из коридора.” Свежий взгляд любого человека на интерфейс лучше, чем взгляд дизайнера потому, что он лишён проклятия знания.

5.2.4 Метод карточной сортировки

Этот узконаправленный метод тестирования позволяет выявить то, как пользователь относит понятия к разным категориям. Создаётся список параметров или подкатегорий, которые пользователь должен классифицировать. Каждый параметр записывается на отдельной карточке. Пользователи группируют карточки наиболее логичным, по их мнению, образом давая группам названия. Наиболее часто используемый способ группировки реализуется в интерфейсе.

Метод обратной карточной сортировки переворачивает задачу. Пользователи по уже созданным группам ищут конкретные карточки.

Вопросы

1. Что такое юзабилити?
2. Что такое UX?
3. Как соотносятся эти понятия?
4. Перечислите и охарактеризуйте эвристики Якоба Нильсона. Приведите примеры и антипримеры.
5. Что такое А/В тестирование?
6. Как можно измерять разницу между вариантами в А/В тестировании?
7. Что такое коридорное тестирование?
8. Опишите метод карточной сортировки и метод обратной карточной сортировки.

Литература

- Якоб Нильсен, Хоа Лоранжер. Web-дизайн: удобство использования Web-сайтов = Prioritizing Web Usability. — М.: «Вильямс», 2007. — С. 368.
- Унгер, Р. UX-дизайн. Практическое руководство по проектированию опыта взаимодействия / Р. Унгер, К. Чендлер. — : Символ-Плюс, 2011. — 336 с.

6 Анализ интерфейса

6.1 Количественная оценка пользовательского интерфейса

Эффективность – один из внешних критериев качества ПО. Можно измерить требуемый объём оперативной памяти и количество процессорного времени при решении программой конкретных задач.

Однако не существует чётких метрик оценки эффективности пользовательского интерфейса. Но существуют модели, которые позволяют оценить отдельные аспекты ПИ с известными упрощениями.

"В сущности, все модели неправильны, но некоторые полезны" – высказывание Джорджа Бокса¹ применимо и к приведённым ниже моделям. Сами по себе они не отражают в точности отдельные аспекты поведения пользователей, но позволяют сделать полезные выводы о проектировании интерфейса или получить сравнительную оценку качества интерфейса.

6.1.1 Закон Фиттса

Закон Фиттса – математическая модель для определения времени указания на цель на экране. Среднее время, затрачиваемое на указание на цель определяется как

$$T = a + b \cdot \log_2 \left(\frac{D}{W} + 1 \right), \quad (6.1)$$

¹Джордж Бокс – статистик, внёсший заметный вклад в такие области, как контроль качества, планирование эксперимента, анализ временных рядов и Байесовский вывод.

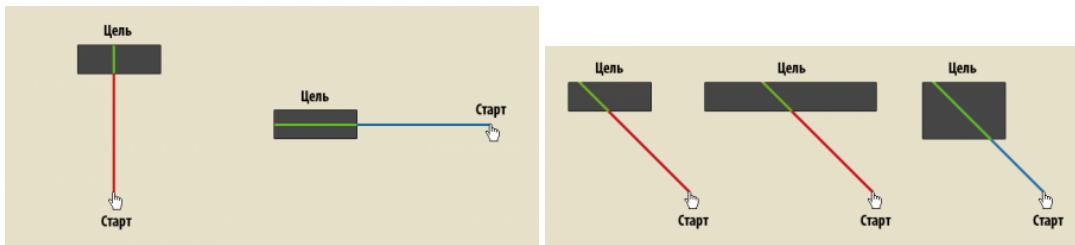


Рисунок 6.1. Закон Фитса: время указания на цель зависит от дистанции до цели (красная и синяя линии) и размера цели вдоль оси движения (зелёная линия). Увеличение ширины цели не всегда даёт выигрыша во времени. Нужно учитывать либо все направления движений, либо самые типичные [62].

где a — среднее время запуска/остановки движения, b — величина, зависящая от типичной скорости движения указателя, D — дистанция от точки старта до центра цели, W — ширина цели, измеренная вдоль оси движения (рис. 6.1).

Значение коэффициентов a и b можно оценить экспериментально, измерив время наведения для разных целей, находящихся на разных расстояниях. Но с точки проектирования интерфейса, важны выводы, следующие из закона, а не конкретное количество времени, которое пользователь затратит на наведение на цель. Тем более что закон был описан Полом Фиттсом в 1954 году в эксперименте, где требовалось указывать стилусом на параллельные друг другу металлические полоски разной ширины (рис. 6.2).

Закон имеет допущения. Пользователь знает где находится указатель, знает где находится цель, на которую он должен указать. Пользователь не промахивается мимо цели. Направление движения курсора не влияет на время указания.

Логарифмическая зависимость позволяет сделать два вывода. Небольшое увеличение небольшой цели делает её простой (быстрой) для указания (наведения). Такое же небольшое увеличение большой цели не даст заметного выигрыша.

Делать все элементы интерфейса большими неразумно. Тем более, если элементы будут находиться слишком близко друг к

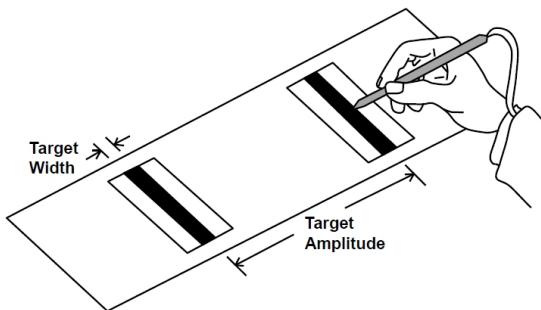


Рисунок 6.2. Оригинальная постановка эксперимента, по результатам которого был описан закон Фиттса. Участник эксперимента должен был как можно быстрее касаться стилусом двух металлических пластин. Эксперимент повторялся для пластин разной ширины и с разным расстоянием между ними [63].

другу, то пользователи будут чаще ошибаться указывая не на тот элемент, который нужно.

Правило размера цели. Нужно увеличивать размер только часто используемых целей. То есть тех кнопок, ссылок, элементов меню и т.д. на которые пользователь во время работы кликает чаще всего.

Правило бесконечной границы. Цели, расположенные у края экрана, имеют условно бесконечную высоту (ширину), так как двигаясь вниз до края курсор не сможет продвинуться дальше границы цели (рис. 6.3). В идеальном случае, как на рисунке, размер верхней цели бесконечен. Значит, логарифм в формуле 6.1 равен нулю. В остальных случаях время указания на цель, помимо расстояния, зависит либо от ширины либо высоты цели.

Цели, расположенные в углах экрана, имеют условно бесконечную ширину и высоту. Поэтому указание на них из любой точки экрана, согласно закону Фиттса, занимает минимально возможное количество времени.

Правило бесконечной границы работает для панели задач ОС Windows (рис. 6.5). Кнопки закрытия окна расположены в правом углу экрана, когда окно развернуто. Вкладки в большинстве браузеров находятся прямо на заголовке окна, то есть

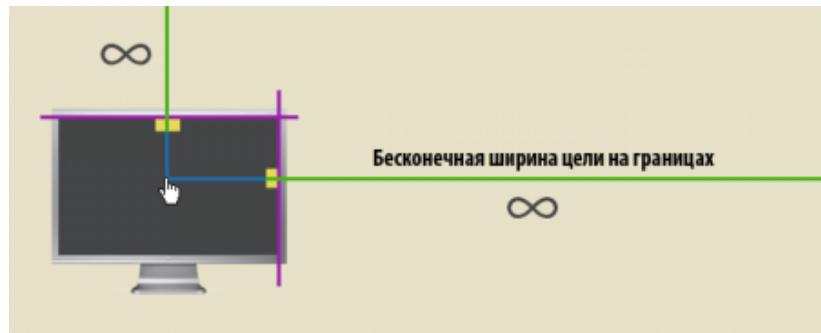


Рисунок 6.3. Правило бесконечной границы. Так как нельзя промахнуться мимо края экрана, ширина или высота цели в направлении границы экрана считается бесконечной.

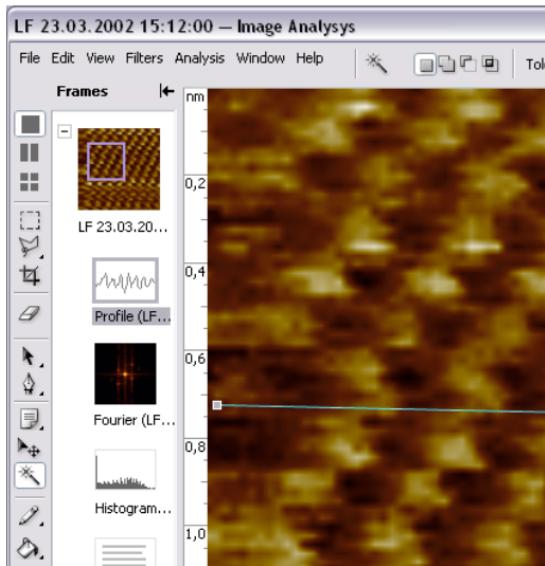


Рисунок 6.4. Правило бесконечной границы: когда окно развернуто во весь экран, кнопки на левой панели инструментов имеют бесконечную ширину (если область нажатия доходит до левой границы окна) [7]

возле верхнего края экрана, когда окно развернуто. Полоса прокрутки во многих приложениях расположена в правой части окна, что даже даёт преимущество, когда окно развернуто.

В семействе операционных систем Mac OS в верхней части экрана традиционно располагается панель меню запущенной программы (рис 6.6).

Элементы, расположенные у границ экрана, удобны ещё и тем, что при указании на них пользователю не приходится замедлять движение курсора, чтобы точнее прицелиться. Что, однако, уже не является следствием закона Фиттса.



Рисунок 6.5. Правило размера цели. Панель задач Windows 7 (сверху), Windows 10 (снизу). Кнопка "Пуск" и кнопка "свернуть все окна" (справа на панели задач) находится в углах экрана, что сокращает время наведения на них до минимально возможного. Круглая кнопка Пуск имеет квадратную область клика.



Рисунок 6.6. MAC OS Leopard. Панель меню запущенной программы всегда находится в верхней части экрана. При наведении на панель программ в нижней части экрана ближайшие к курсору кнопки увеличиваются, облегчая наведение.

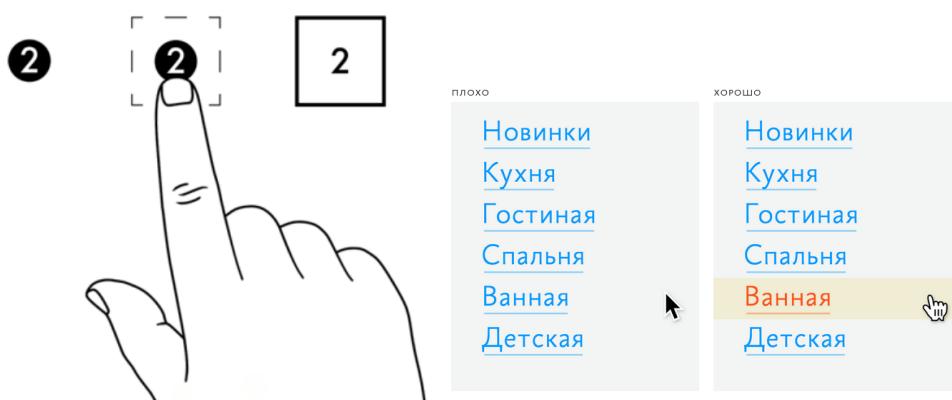


Рисунок 6.7. Если объект нельзя увеличить в размерах, то можно попробовать увеличить невидимую область нажатия вокруг него. Примеры из bureau.ru/books/ui/demo/4

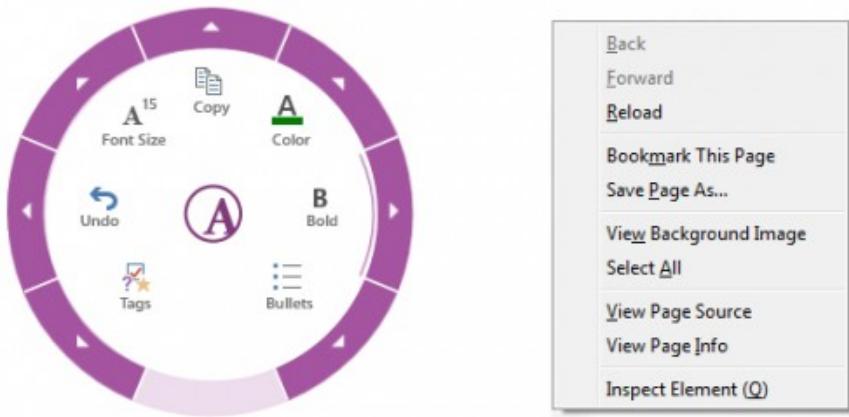


Рисунок 6.8. Круговое контекстное меню. Время указание на все элементы, в отличие от классического контекстного меню, одинаково, а сами элементы больше. Классическое контекстное меню. Самые часто используемые пункты должны быть расположены ближе к указателю мыши (выше)

Кроме того, часто используемые элементы логично делать большими ещё и потому, что они становятся заметнее. Уменьшится время поиска цели. *todo:*

Минимизировать время указания можно и минимизируя расстояния до цели, в формуле закона Фиттса (6.1). Контекстное меню всегда открывается в месте клика, так что не приходится слишком далеко вести курсор для выбора пункта. Самые часто используемые элементы меню стоит помещать в самый верх, ближе к начальной позиции курсора. В круговом контекстном меню оптимизируется и расстояние до всех элементов их размер.

Если элементы интерфейса могут быть выбраны последовательно, то, возможно, стоит расположить их ближе друг к другу (рис 6.9).

С другой стороны, делайте достаточное *расстояние* до кнопок, в которые лучше случайно не нажимать

Закон Фиттса подразумевает, что пользователь знает, где находится элемент интерфейса, на который он хочет навести курсор. Пользователь не знает, где искать элемент, то ко времени

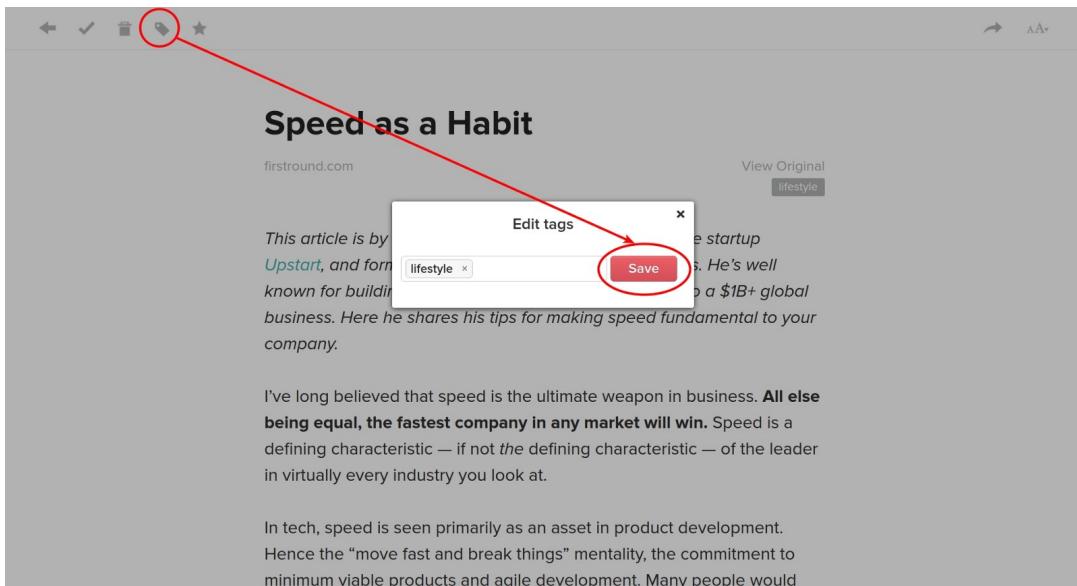


Рисунок 6.9. Ожидается, что кнопки должны быть нажаты последовательно, однако они разнесены на большое расстояние.

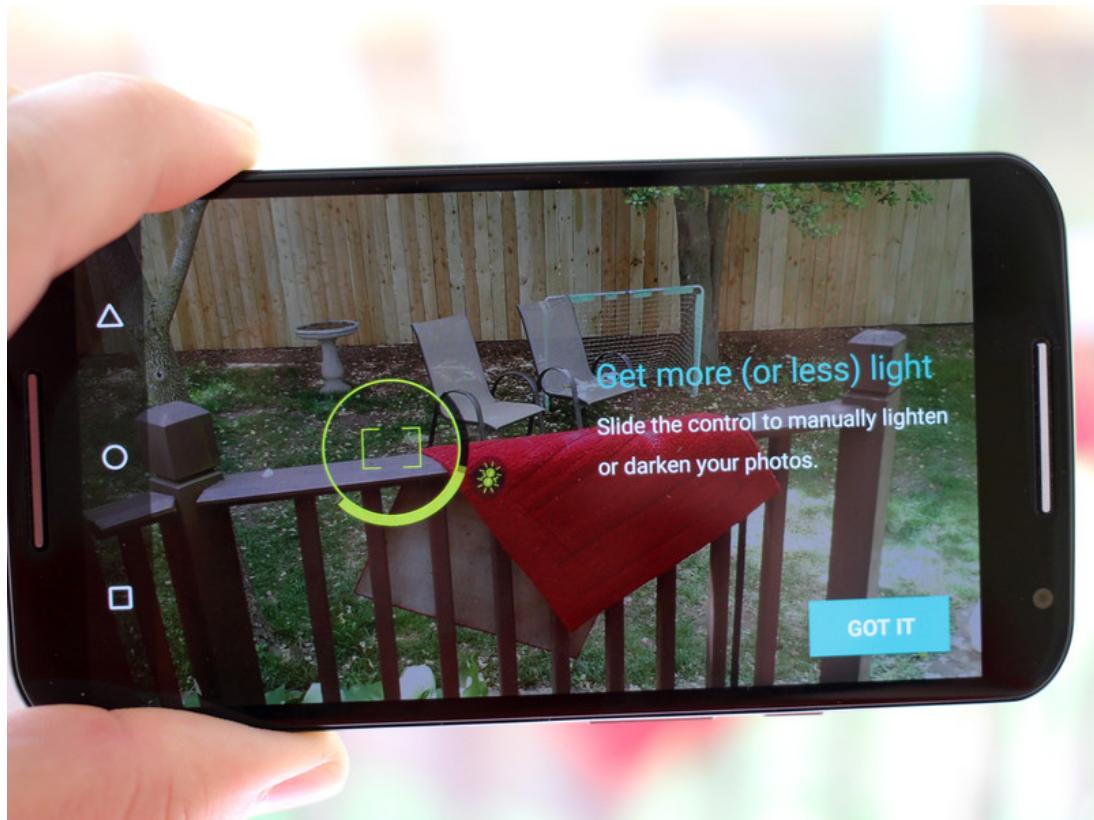


Рисунок 6.10. Расстояние до цели равно нулю. Кнопка появляется в месте тапа.

позиционирования добавляется время поиска этого элемента на экране. Это тоже нужно минимизировать.

Типичные элементы управления следует располагать там, где пользователи ожидают их увидеть. Например, кнопка входа на сайт, как правило, располагается в верхней правой части сайта, меню "файл" стоит в начале панели меню, а пункт "копировать" — самый первый в контекстном меню для текста.

Скорее всего у пользователей сформированы привычки, для взаимодействия с типичными элементами интерфейса. Ко времени наведения не будет добавлено время поиска элемента, а само взаимодействие с ним с высокой долей уверенности будет автоматичным.

6.2 Закон Хика и проблемы выбора

Время, затрачиваемое на выбор, является функцией числа альтернатив

$$T = a + b \cdot \log_2 (n + 1), \quad (6.2)$$

где T — значение времени реакции, усреднённое по всем альтернативам; a, b — константы; n — число равновероятных альтернативных альтернатив; $+1$ — дополнительная альтернатива — случай отказа от выбора.

Под альтернативами можно понимать различные варианты действия пользователя. Например, выбор нужного элемента меню, вкладки, кнопки на панели инструментов.

Коэффициенты a и b , используемые в выражении закона Хика, в большой степени зависят от многих условий, включая то, как представлены возможные варианты, и то, насколько хорошо пользователь знаком с системой. Наличие навыков и привычек в использовании системы снижает значение b .

Как и для закона Фиттса логарифмическая зависимость поз-

воляет сделать вывод: немного уменьшая небольшое количество альтернатив мы существенно снижаем время выбора, такое же уменьшение большого числа альтернатив не даёт заметного выигрыша во времени.

Закон имеет допущения и ограничения. Пользователь знает обо всех доступных действиях, среди которых должен сделать выбор, а не узнаёт о них просматривая, например, контекстное меню. Закон не применяется, если выбор связан с принятием сложного решения, изучением каждого из вариантов и т.д. Например, выбор из 5 автомобилей для покупки и выбор из 5 супов в ресторане, скорее всего, займут разное количество времени. Закон Хика не описывает время *поиска* элемента на экране, среди множества других.

Для случая, когда вероятности выбора альтернатив существенно отличаются, используется следующая формула

$$T = a + b \sum_i^n p_i \log_2(1/p_i + 1), \quad (6.3)$$

где p_i – вероятность выбора i -й альтернативы.

Из формулировок закона следуют простые выводы. Сокращайте число альтернатив. Если это сделать невозможно, учитывайте то, что некоторые варианты могут выбираться чаще чем другие. Спрятите часть непопулярных вариантов в подменю или подкатегории². Например, если категорий товаров очень много и часть из них весьма специфические, то показывайте только самые популярные категории, добавьте кнопку "Показать все категории на сайте интернет-магазина. При такой организации выборы в большинстве случаев, пользователи смогут сделать быстрый выбор популярных вариантов среди небольшого числа и только изредка им придётся выбирать из большого числа вариантов или делать это в два этапа.

²о том, как придумать перечень категорий или разделов, например, на сайте, читайте в параграфе про метод карточной сортировки

Делайте выбор за пользователя, если это возможно³. Например, предлагайте ему популярные товары или рекомендуйте их, основываясь на предпочтениях конкретного пользователя.

Сам процесс выбора из большого числа альтернатив может вообще отталкивать пользователей от совершения выбора (+1 в формуле 6.2. Тогда, сократите сложность принимаемого решения разбив его на несколько этапов. А если же пользователь не может отказаться от выбора, это требует от него лишних ментальных усилий. Что может ухудшить пользовательский опыт.

Ко времени непосредственно выбора может добавиться время, необходимое на изучение перечня альтернатив и время поиска элемента интерфейса, отвечающего за альтернативу на экране.

Время поиска элемента можно сократить, если, опять же, сперва показать самые вероятные альтернативы. Например, при регистрации на сайте нужно указать страну проживания или гражданство, то можно показать самые популярные варианты в начале списка. Можно сделать выбор за пользователя, основываясь на географической привязке его IP адреса. Если эти варианты всё же не сработают, то пользователь должен понимать принцип организации списка. Например, список стран будет отсортирован по алфавиту. Тогда пользователь будет иметь примерное представление где искать нужный элемент списка.

6.3 GOMS

Для оценки времени, которое требуется пользователю, для решения задачи с помощью программы используется модель GOMS.

Модель GOMS (the model of Goals, Objects, Methods, and Selection rules, правила для целей, объектов, методов и выделения) позволяет предсказать время, необходимое для выполнения зада-

³см. также параграф про информационную эффективность интерфейса

чи с помощью конкретного интерфейса.

Весь процесс взаимодействия пользователя устройствами ввода разбивается на элементарные жесты.

Время, требующееся для выполнения какой-то задачи системой «пользователь – компьютер», является суммой всех временных интервалов, которые потребовались системе на выполнение последовательности элементарных жестов, составляющих данную задачу.

Для большей части сравнительного анализа задач, включающих использование клавиатуры и графического устройства ввода, вместо проведения измерений для каждого отдельного пользователя можно применить набор стандартных интервалов.

Эта модель, как и законы Фиттса и Хика, даёт *сравнительную* оценку интерфейсов.

Элементарные Жесты

- **K** = 0.2 с. Нажатие клавиши.
- **P** = 1.1 с. Указание. Время, необходимое для того, чтобы указать на позицию на мониторе.
- **H** = 0.4 с. Перемещение руки с ГУВ⁴ на клавиатуру и обратно.
- **M** = 1.35 с. Ментальная подготовка. Время необходимое чтобы умственно подготовиться к следующему шагу.
- **R** Ответ. Ожидание ответа компьютера⁵.

Ограничения модели GOMS

- Рассматривается средний пользователь
- Пользователь знаком с интерфейсом

⁴графическое устройство ввода данных, например, мышь.

⁵анализ времени отклика компьютеров: 1977-2017

- Не учитываются размеры и положение элементов интерфейса
- На учитывается сложность принятия решения в ментальной операции
- Характеристики устройств ввода не влияют на время жестов

Вычисления времени, необходимого на выполнение действия, с помощью модели GOMS начинаются с перечисления операций из списка жестов модели GOMS. Затем расставляются ментальные операторы согласно правилам. Потом, ментальные операторы удаляются согласно правилам.

Расстановка ментальных операций.

- **Правило 0.** Начальная расстановка операторов M

Операторы M следует устанавливать перед всеми K (нажатие клавиши) и P (указание с помощью ГУВ), предназначеными для выбора команд; но перед операторами P, пред назначенными для указания на аргументы этих команд, ставить оператор M не следует.

- **Правило 1.** Удаление ожидаемых операторов M

Если оператор, следующий за оператором M, является полностью ожидаемым, с точки зрения оператора, предшествующего M, то этот оператор M может быть удалён.

Например, если вы перемещаете ГУВ с намерением нажать его кнопку по достижении цели движения, то в соответствии с этим правилом следует удалить оператор M, устанавливаемый по правилу 0. В этом случае последовательность РМК превращается в РК.

- **Правило 2.** Удаление операторов M внутри когнитивных единиц

Если строка вида М К М К М К... принадлежит когнитивной единице, то следует удалить все операторы М, кроме первого.

Когнитивной единицей является непрерывная последовательность вводимых символов, которые могут образовывать название команды или аргумент. Например, «Елена Троянская» или «-4564.23» являются примерами когнитивных единиц.

- **Правило 3.** Удаление операторов М перед последовательными разделителями

Если оператор К означает лишний разделитель, стоящий в конце когнитивной единицы (например, разделитель команды, следующий сразу за разделителем аргумента этой команды), то следует удалить оператор М, стоящий перед ним.

- **Правило 4.** Удаление операторов М, которые являются прерывателями команд

Если оператор К является разделителем, стоящим после постоянной строки (например, название команды или любая последовательность символов, которая каждый раз вводится в неизменном виде), то следует удалить оператор М, стоящий перед ним. (Добавление разделителя станет привычным действием, и поэтому разделитель станет частью строки и не будет требовать специального оператора М.) Но если оператор К является разделителем для строки аргументов или любой другой изменяемой строки, то оператор М следует сохранить перед ним.

- **Правило 5.** Удаление перекрывающих операторов М

Любую часть оператора М, которая перекрывает оператор R, означающий задержку, связанную с ожиданием ответа компьютера, учитывать не следует.

Пример оценки интерфейса.



Рисунок 6.11. Окно авторизации в Microsoft Outlook. Пример для оценки времени решения задачи пользователем по модели GOMS.

Задача пользователя: ввести логин (5 символов) и пароль (5 символов) и нажать OK.

Взаимодействовать с формой авторизации пользователи могут по-разному. Переходить от одного поля ввода к другому с помощью клавиши *tab* или клика мыши. Здесь рассмотрим вариант, когда пользователь не знаком с горячими клавишами (рис. 6.12). Исходное положение: рука пользователя лежит на мыши (после запуска программы), поле ввода логина уже в фокусе ввода. Сравнить среднее время с другим вариантом взаимодействия с интерфейсом предлагается читателю.

Некоторые способы ввода данных с точки зрения GOMS занимают примерно одинаковое время. Например, задание значения числового параметра с помощью ползунка займёт 2.4 секунды (РКРК): навести указатель, нажать клавишу мыши, навести на новое значение, отпустить клавишу. Ввод трёхзначного числа в поле клик по поле ввода займёт 2.3 секунды (РК-Н-ККК). Но оперировать небольшим ползунком, который перемещается вдоль одной оси проще чем, указывать на любую другую цель на экране. Фактически это будет занимать почти всегда меньше времени чем 1.1 с. С другой стороны, точно указать значение с помощью ползунка быстро вряд ли можно.

Иногда стоит комбинировать способы ввода данных, чтобы

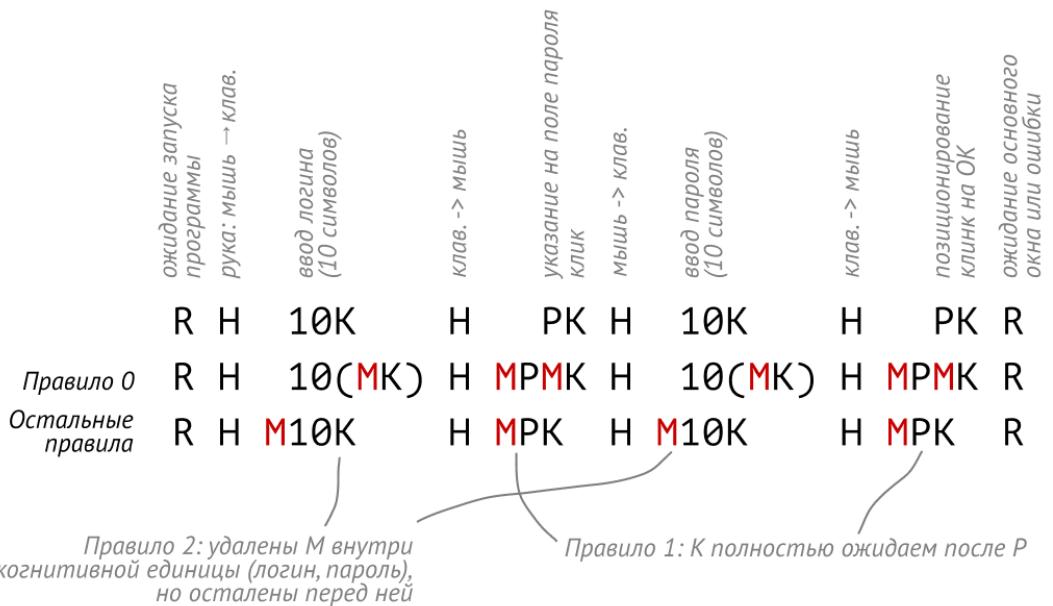


Рисунок 6.12. Начальная запись жестов модели GOMS для окна ввода 6.11; применение правила 0 для расстановки когнитивных операторов; удаление когнитивных операторов по правилам 1-5.

пользователь мог выбрать наиболее удобный для него способ взаимодействия (рис. 6.13).

6.4 Информационная эффективность интерфейса

6.4.1 Информативность интерфейса

GOMS и закон Фиттса оценивают время на совершения операций. Другая характеристика пользовательского интерфейса – количество информации, которое вводит пользователь.

Информативность – это доля смысловой части в общей длине сообщения.

Интерфейс предполагает обмен информацией, а пропускная способность любого канала ограничена. Поэтому высокая информативность – признак хорошего интерфейса.

Повышать информативность можно двумя способами: убирая лишние или пряча редко используемые элементы интерфейса (рис. 6.14), создавая меньше "визуального мусора" и добавлять смысловую нагрузку существующим элементам интер-

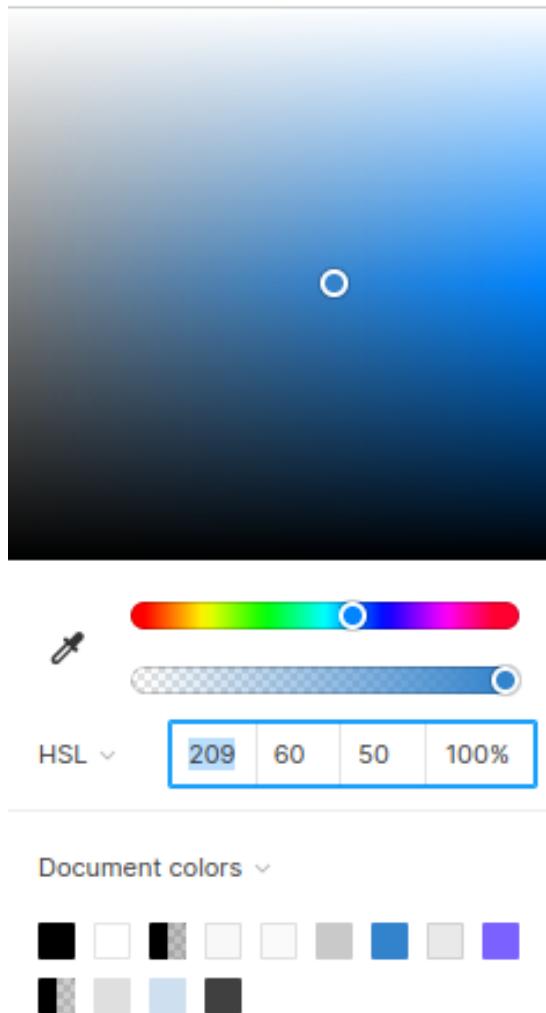


Рисунок 6.13. Выбор цвета в Figma. Пользователь может воспользоваться ползунком для быстрого выбора оттенка (Hue) и палитрой для выбора насыщенности и светлоты (saturation и value). Для точного задания этих параметров можно ввести значения в поля ввода. Список использованных ранее цветов (document colors) вообще избавляет от процедуры задания всех параметров цвета по отдельности.

файса (рис. 6.15).

Один из приёмов повышения информативности – вынос за скобки. Повторяющуюся информацию (слова, пиктограммы и т.п.) в перечных приводят один раз, вынося на один уровень иерархии выше (рис. 6.16, 6.16). На рис. 3.9 первая колонка таблицы содержит повторяющиеся значения. В улучшенной версии таблицы (рис. 3.10) каждое значение проводится один раз, одновременно являясь как бы заголовком к остальному содержимому.

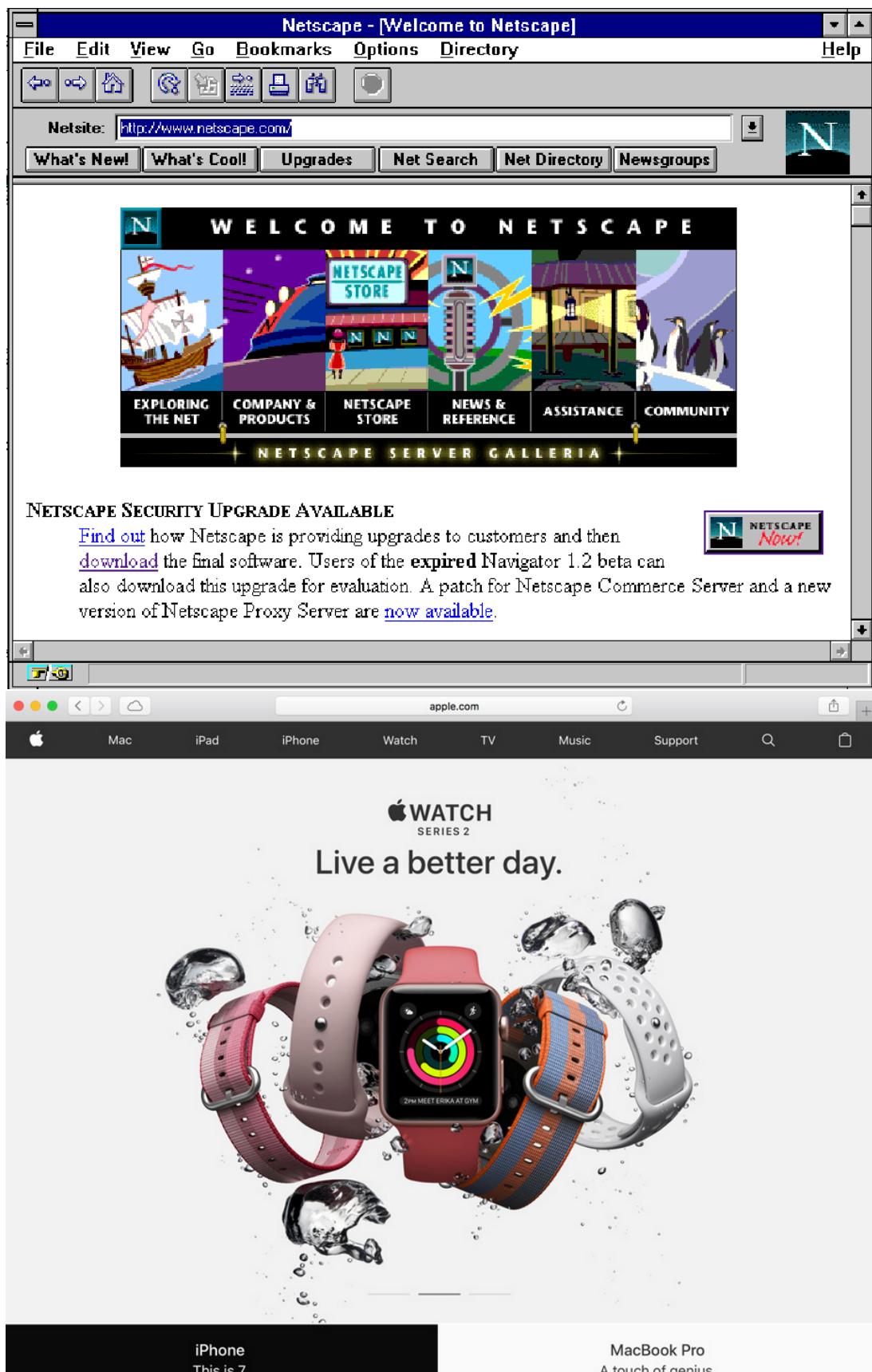


Рисунок 6.14. Интерфейс старых браузеров содержал множество видимых элементов интерфейса (вверху, браузер NetScape Navigator). Современные браузеры, при возросшем разрешении дисплеев, (внизу, Safari, 2017) скрывают меню, многие кнопки скрыты или расположены компактно вместе с адресной строкой, полоса прокрутки и строка состояния появляются только когда это необходимо



Рисунок 6.15. Примеры повышение информативности интерфейса: краткое описание изменений в сообщении о новой версии программы. Объединение кнопки "Купить" и цены товара.

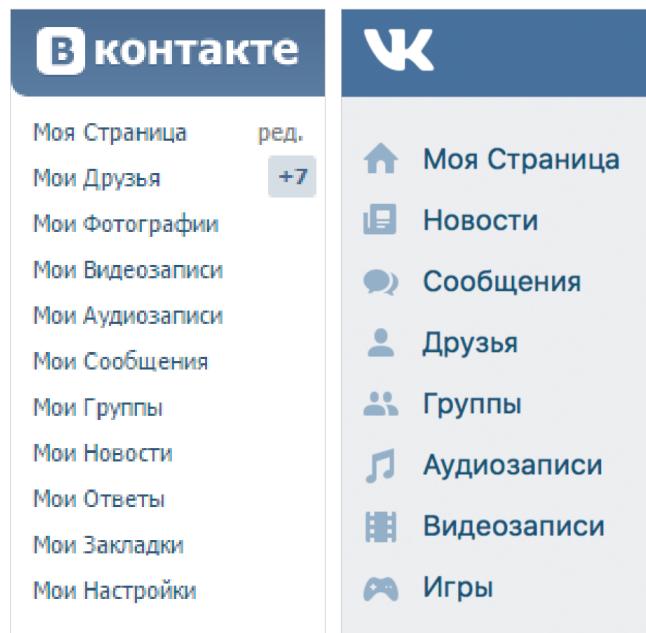


Рисунок 6.16. Слово "Мои" можно "вынести за скобки".

Поискать в регионах: [СНГ](#); [Россия](#) (Красноярск, Томск, Новосибирск, Кемерово)

Поискать в рубрике: [Культура и искусство](#) (Русская проза, Библиотеки, Живопись)
[Общество и политика](#)

Поискать то же самое на: [AltaVista](#) · [Google](#) · [MSN](#) · [Yahoo](#)

Поискать то же самое

в регионе: [СНГ](#); [Россия](#) (Красноярск, Томск, Новосибирск, Кемерово)

в рубрике: [Культура и искусство](#) (Русская проза, Библиотеки, Живопись)
[Общество и политика](#)

в других поисковых системах: [AltaVista](#) · [Google](#) · [MSN](#) · [Yahoo](#)

«Пушкин»

в регионе: [СНГ](#); [Россия](#) (Красноярск, Томск, Новосибирск, Кемерово)

в рубрике: [Культура и искусство](#) (Русская проза, Библиотеки, Живопись)
[Общество и политика](#)

в других поисковых системах: [AltaVista](#) · [Google](#) · [MSN](#) · [Yahoo](#)

Рисунок 6.17. Улучшения предложения продолжить поиск (верхний вариант, такой блок приводился ниже результатов поиска на странице yandex.ru) в два этапа: вынос за скобки "Поискать замена обозначения данных "то же самое" на сам поисковый запрос

6.4.2 Оценка информативности.

Информационная производительность(эффективность) интерфейса Е – это отношение минимального количества информации I_{\min} , необходимого для выполнения задачи, к количеству информации $I_{\text{введ.}}$, которое фактически требуется ввести от пользователя.

$$E = \frac{I_{\min}}{I_{\text{введ.}}} \quad (6.4)$$

Дополнительно рассмотрим пограничные случаи, не учитываемые формулой 6.4. Если никакой работы для выполнения задачи не требуется или работа просто не производится, то производительность составляет 1. Если действия не требуется, но оно производится, то производительность составляет 0 (рис. 6.18).

Количество вводимой информации, когда пользователь совершают выбор из n альтернатив, определяется выражением:

$$I = \log_2(n)$$

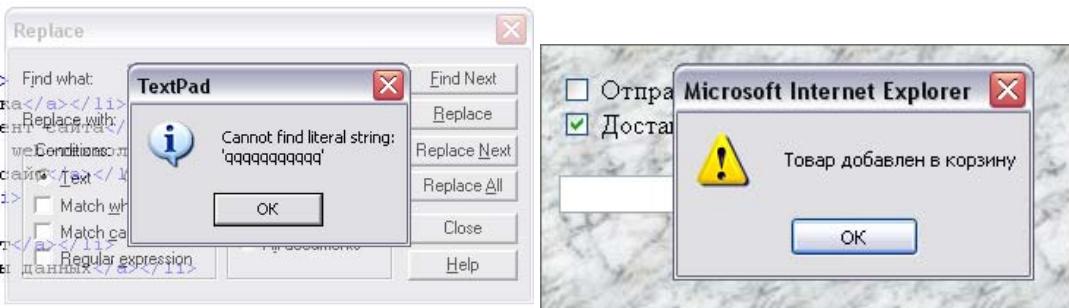


Рисунок 6.18. Нулевая информационная эффективность: данных для ввода не требуется, но пользователь должен ввести один бит информации нажав на кнопку Ok

При этом отказ от выбора не рассматривается.

Если же альтернативы не равновероятны, информация, передаваемая i -й альтернативой, определяется по формуле

$$I = p_i \log_2(1/p_i)$$

где p_i - вероятность выбора i -й альтернативы.

Символьная эффективность определяется как минимальное количество символов, необходимое для выполнения задачи, отнесённое к количеству символов, которое в данном интерфейсе требуется ввести пользователем.

Вопросы

1. Сформулируйте закон Фиттса. Какие величины в него входят?
2. Какие условия применения закона?
3. Какие рекомендации следуют из закона Фиттса?
4. На какие 5 пикселей на экране можно кликнуть быстрее всего?
5. Как влияет расположение элемента возле краёв экрана на время указания на него?
6. Сформулируйте закон Хика. Какие выводы из него следуют?

ют?

7. Как уменьшить время поиска или выбора элемента в списке?
8. Для чего используется метод GOMS?
9. Какие элементарные жесты входят в модель GOMS?
10. Назовите правила записи жестов.

Дополнительная литература

- Раскин Д., Интерфейс: новые направления в проектировании компьютерных систем. – Пер. с англ. – СПб: Символ-Плюс, 2007. – 272 с.
- Бирман И. Б. Пользовательский интерфейс. — М.: Изд-во Бюро Горбунова, 2017. Ознакомительный фрагмент книги: <https://bureau.ru/books/ui/demo/>

7 Типографика и тексты

Текст – это около 90% всей визуальной информации, которую получает человек. Большая часть информации в пользовательских интерфейсах – тоже текст (рис. 7.1). Поэтому важно понимать, как доносить информацию через текст эффективно.

Представления о способах передачи текстовой информации несколько менялись вместе с технологиями тиражирования и отображения текстовой информации. Начиная со времени когда стали широко доступны печатные машинки, а потом матричные принтеры в широкой практике сложились определённые привила структурирования текстовой информации. Которые, из-за скучных технических возможностей печатающих устройств, были актуальны для своего времени. Но некоторые из них, вроде частого выделения важных слов в тексте подчёркиванием, сохранились до сих пор. При этом широкий круг возможностей электронной типографики часто игнорируется. Для выделения важного часто используется подчёркивание, полужирное начертание, верхний регистр, яркий красный цвет, но реже просто увеличение размера, смена цвета на заметный, но гармонирующий с остальным текстом. Заголовки, пояснения и другие особые элементы текста иногда никак не выделяются, создавая эффект стены текста. С другой стороны, многие возможности используются во вред. Часто случается неуместное использование многих узконаправленных или специализированных шрифтов (Comis Sans, готические и рукописные шрифты), неудачная комбинация большого количества шрифтов, концентрация на эстетической составляющей в ущерб чи-

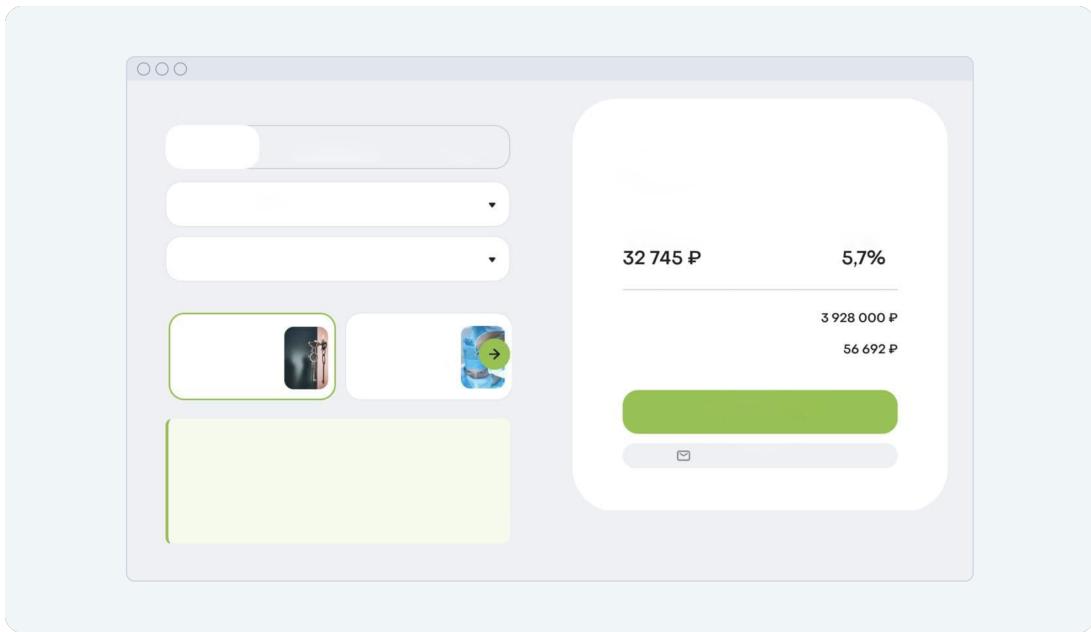


Рисунок 7.1. Графический пользовательский интерфейс без слов чаще всего невозможен

таемости.

Типографика – это технология и искусство делать текст различимым (legibility), удобочитаемым (readability) и эстетичным (aesthetic).

Типографика, с одной стороны, представляет собой одну из отраслей графического дизайна, с другой – свод строгих правил, определяющих использование шрифтов для создания наиболее понятного для восприятия читателя текста.

Рассмотрим основные понятия типографики и приведём классификацию шрифтов.

7.1 Классификация шрифтов

В первую очередь шрифты можно разделить (см. рис 7.3) на две категории – шрифты с засечками (serif, антиква) и шрифты без засечек (sans serif, готеск)

Нет убедительных доказательств того, что антиква или готеск должны использоваться в отдельных ситуациях. Часто рекомендуют использовать шрифт с засечками для больших тек-

Legibility
is how well you
see the letters.

Readability

is how easily you read the words, as in long passages of text. there are very different requirements in each case, depending on the visibility of the text and the level of experience of the reader.

Рисунок 7.2. Различимость (legibility) – свойство символов быть отличными друг от друга; читаемость (readability) – свойства текста, характеризующее простоту восприятия читателем, зависящую в частности от свойств шрифта

Рисунок 7.3. Гротеск – sans serif – шрифт без засечек. Антиква – serif – шрифт с засечками. Антиквенные шрифты, как правило, имеют разную толщину основного и дополнительно штриха, в отличие от гротеска.

Helvetica	TRAJAN
Arial	Baskerville
Roboto	Times New Roman
San Francisco	Georgia
PT Sans	PT Serif

Рисунок 7.4. Слева – шрифты с засечками, справа – без засечек.

Handgloves

Handgloves

Handgloves

Рисунок 7.5. Антиква (вверху) и её подвид – брусковый шрифт (внизу), в середине – шрифт имеющий черты антиквы и брускового.

стов. Например, в оформлении документов, книг, текстовых статей на сайтах. Шрифт без засечек рекомендуют для отдельных, коротких надписей. Иногда рекомендуют использовать антикву для печатных текстов, готеск – для электронной типографики. Эти рекомендации имеют много исключений и чаще описывают традицию, не основываясь на исследованиях.

Брусковый шрифт – шрифт с мощными засечками прямоугольной формы без скруглений или с небольшим скруглением в местах присоединения к основным штрихам (см. рис ??). В таких шрифтах толщина основных и дополнительных штрихов отличается меньше чем в антикве.

Шрифты этого типа занимают ведущее место по шкале читабельности и идеально подходят для набора длинных текстов, так как имеют очень слабый контраст. Однако страница, набранная этим шрифтом, выглядит значительно «темнее» страницы, набранной обычной гарнитурой, поскольку штрихи брускового типа плотнее и более единообразны по толщине. Брусковый шрифт часто используется при наборе детских книг.

Акцидентные (декоративные) шрифты – самая большая группа шрифтов, непохожих друг на друга. Такие шрифты не предназначены для набора основного текста и используются для заголовков, небольших отрывков текста с целью привлечения и акцентирования внимания или служат исключительно эсте-



Рисунок 7.6. Акцентные шрифты своим характерным видом часто призваны подчеркнуть общий стиль дизайна.

Proportional
Monospaced

Рисунок 7.7. Пропорциональный (proportional) и моноширинные (monospaced) шрифты.

тическим целям (рис. 7.6). Эти шрифты обязательно содержат начертания для всех символов и нередко создаются прямо во время работы над общим дизайном продукта, например, визитки или логотипа.

Более формальный подход к классификации шрифтов – учить различия в ширине символов (рис. ??):

- Пропорциональные;
- Моноширинные.

Символы пропорциональных шрифтов имеют ширину, пропорциональную их конструкции. Например, буква ж шире чем к. В моноширинных шрифтах все символы имеют одинаковую ширину.

Используйте моноширинные шрифты когда: важно вырав-

Roboto
1 1 1 | 0 0 0

Source code pro
1 I l | 0 0 o

Рисунок 7.8. В моноширинных шрифтах (например, Source Code Pro), часто уделено много внимания различимости похожих символов. В пропорциональных шрифтах (например, Roboto) бывают очень похожие символы.



Рисунок 7.9. Моноширинные шрифты используются в редакторах кода, где важно выравнивание кода. Некоторые моноширинные шрифты имеют лигатуры – отдельные начертания, обозначающие популярные сочетания символов, например `->`. Слева – шрифт Consolas, справа – шрифт Fira Code

нивание символов (см. рис. 7.9) и различимость (см. рис. 7.8) символов. Некоторые шрифты имеют режим отображения цифр с подстройкой одинаковой ширины (OpenType feature – Tabular Figures). Последнее важно, например, для отображения числовых данных в колонке таблицы так, чтобы разряды находились друг под другом (см. рис. 3.10).

Наконец приведём наименее формальную классификацию шрифтов: нейтральные и характерные. Нейтральные шрифты не выделяются, у них нет заметных особенностей. Часто они выглядят похожими друг на друга. Эти шрифты часто служат утилитарной цели – доносить информацию, не привлекая внимания к особенностям начертания символов. Характерные шрифты непохожи на другие, их легче различать. Для них важно узнавание, поэтому их часто разрабатывают как часть экосистемы бренда.

Typeface vs Font

Typeface: Helvetica

Font: Helvetica Light
Helvetica Regular
Helvetica Oblique
Helvetica Bold
Helvetica Bold Oblique

Рисунок 7.10. Шрифт (font) и гарнитура (typeface)

7.2 Основные понятия из типографики

Шрифт (font) — графический рисунок начертаний букв и знаков, составляющих единую стилистическую и композиционную систему, набор символов определённого размера и рисунка. Гарнитура (typeface) — набор из одного или нескольких шрифтов в одном или нескольких размерах и начертаниях, имеющих стилевое единство рисунка и состоящих из определённого набора типографских знаков (см. рис 7.10).

Одна гарнитура может содержать несколько начертаний отличающихся начертаний по

- толщине (см. рис 7.11);
- стилю: прямое начертание (roman), наклонное (oblique), курсив (italic), см. рисунок 7.13;
- плотности начертания: обычная (normal), плотное (condensed), разреженное (extended) и др., см. рис ??.

Наклонное начертание часто представляет собой символы прямого начертания, только с наклоном, в отличие от курсива, где символы могут иметь совершенно другое начертание. Это хорошо видно на примере буквы а, на рисунке 7.13.

Не все шрифты содержат все перечисленные виды начертаний. Если в шрифте нет курсивного начертания, то оно часто

Thin
Extra-Light
Light
Regular
Medium
Semi-Bold
Bold
Extra-Bold
Black

Рисунок 7.11. Гарнитура Roboto, начертания с толщинами от 100 до 900

Normal - Sphinx of black quartz, judge my vow.

Italic - Sphinx of black quartz, judge my vow.

Oblique - Sphinx of black quartz, judge my vow.

Рисунок 7.12. прямое начертание (roman), наклонное (oblique), курсив (italic)

заменяется наклоном символов без изменения начертания.

Наконец шрифты отличаются набором символов. Например, многие латинские шрифты не содержат начертаний для кириллических алфавитов. Многие алфавиты для китайского языка содержат всего один вид начертания латинского алфавита.

Разработка шрифта – долгий процесс, где требуется нарисовать много символов. Поэтому, часто разработчики ограничиваются малым числом начертаний, исключают курсив. Например, шрифт Roboto, разработанный компанией Google, имеет 12 начертаний отличающихся толщиной (100, 300, 400, 500, 700, 900) и начертанием (regular и italic), а шрифт PT Sans, разработанный компанией ParaType имеет 4 начертания.

Разработанный шрифт помимо самих начертаний ещё содержит информацию о сочетаниях символов. Кернинг (kerning) – избирательное изменение интервала между буквами в зависи-

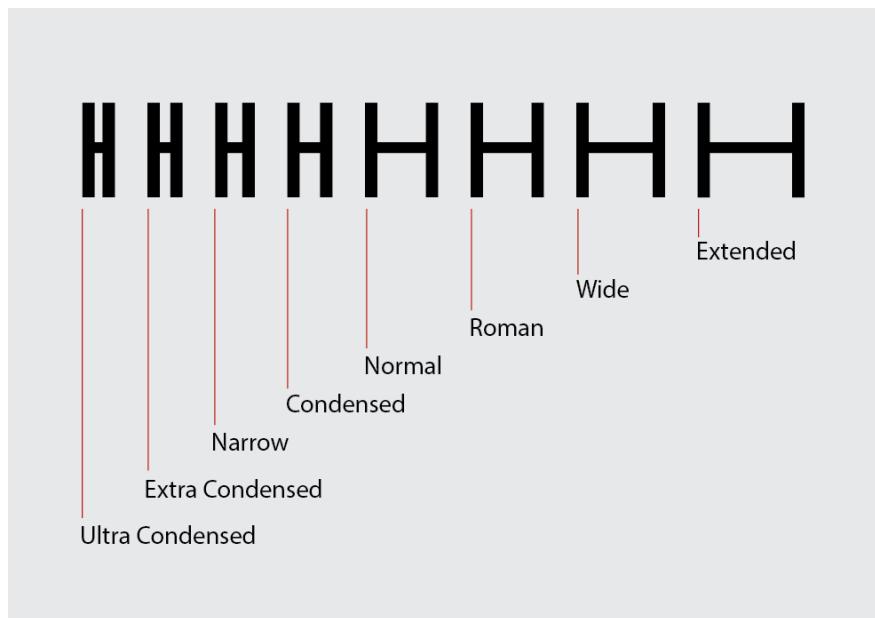


Рисунок 7.13. Различные плотности начертания



Рисунок 7.14. Наличие равномерного расстояния между символами в слове не означает, что и визуальное (воспринимаемое человеком) расстояние будет таким же

симости от их формы.

Проблема создания большого числа начертаний может быть частично решена автоматической настройкой параметров шрифта (например, толщины и плотности) по заданным разработчиками шрифта правилам. Такие шрифты называются переменными (variable). Примеры можно увидеть на сайте: v-fonts.com (рис. ??)

7.2.1 Выбор шрифта

Подбор шрифта или сочетания шрифтов – это творческая задача, плохо поддающаяся формализации. Но можно выделить отдельные рекомендации.



Helvetica Now Variable

ABCDEFGHIJKLMNOPQRSTUVWXYZ
abcdefghijklmnopqrstuvwxyz 012345
6789

Рисунок 7.15. Фрагмент веб-страницы (v-fonts.com) с демонстрацией переменного (Variable) шрифта Helvetica Now

Во-первых, шрифт должен хорошо читаться в предполагаемых условиях использования. В Некоторых случаях особенно важна хорошая различимость символов. Например, если шрифт будет использоваться для отображения логинов, паролей, исходного кода на языке программирования. Многие гарнитуры специально создаются для определённых целей и носителей.

Например, Roboto, San Francisco и Segoe созданы специально для отображения на дисплеях. Они содержат множество начертаний, поэтому всего одна гарнитура может быть использована для всех элементов пользовательского интерфейса (рис. 7.16). Шрифт Comic Sans был разработан для текста показываемого в «облачках слов» виртуальных помощников в Microsoft Bob (рис. 2.7). Используемый ранее Times New Roman по стилю плохо подходил для этих целей.

Во-вторых, шрифт должен соответствовать общему дизайну веб-страницы или приложения. Нейтральные шрифты могут быть использованы практически везде. Шрифт может быть описан в руководстве по дизайну (Human Interface Guidelines [31, 32, 33, 34, 35, 36]) для операционной системы или платформы, брендбуке компании или руководстве по фирменному стилю.

Headline 1

Headline 2

Headline 3

Headline 4

Headline 5

Headline 6

Subtitle 1

Subtitle 2

Body 1

Body 2

BUTTON

Font	Weight	Size	Letter spacing
Roboto	Medium	14px	1.25px

Caption

OVERLINE

Рисунок 7.16. Рекомендации Material Design по использованию гарнитуры Roboto для разных элементов интерфейса

КОФЕ
30 руб.

К О Ф Е
300 ₽

Рисунок 7.17. Шрифт может существенно влиять на восприятие продукта, подчёркивать его свойства. Times New Roman – шрифт используемый по умолчанию во многих ситуациях, для документов и небрежно сделанных объявлений. В этом случае он подходит скорее для подчёркивания своего ценника.

Наконец, самый субъективный способ: опишите бренд, сайт или текст набором прилагательных, подберите шрифт, которому также подходят эти прилагательные.

7.2.2 Шрифтовая иерархия

Иерархия в типографике – способ показать отношение между отдельными частями текста. Например, заголовки имеют больший по отношению к основному тексту размер.

Иерархия даёт понять:

- структуру сообщения:
заголовок, подзаголовок, краткое содержание, основное сообщение, второстепенная информация и т. д.
- отношение между текстовыми блоками.

Исследования [65] показывают, что больше половины пользователей просматривают веб-страницы, но не читают их содержимое последовательно и целиком. Для таких пользователей важно быстро понять структуру страницы или сообщения, находить важную и интересную для них информацию. Правиль-но выстроенная шрифтовая иерархия облегчает сканирующее чтение (рис. 7.21).

Иерархию можно показывать используя

Курсовая работа по программированию
«База данных библиотеки»

Курсовая работа по программированию
«База данных библиотеки»

Курсовая работа по программированию
«База данных библиотеки»

*Курсовая работа по программированию
«База данных библиотеки»*

Рисунок 7.18. Шрифт Пушкин (внизу) – слишком характерный для документа, с плохой удобочитаемостью и различимостью. Courier New (второй снизу) наоборот обладает хорошей различимостью, но удобочитаемость моноширинных шрифтов для больших текстов сравнительно невысокая. Comic Sans (второй сверху) выглядит слегка небрежно для такого документа, буквы неровные, будто написаны от руки. PT Serif – шрифт с засечками, такие традиционно используются в документах, выглядит достаточно нейтрально для документа.

- Размер шрифта
- Отступы
- Цвет текста
- Толщину, начертание (курсив, наклонный, обычный)
- Прописные, строчные буквы, капитель
- Трекинг
- Гарнитуру

7.2.3 Выравнивание текста и длина строки.

При выравнивании текста по ширине лучше избегать появления непропорционально длинных пробелов. Выравнивание по левому краю лучше выравнивания по ширине, если возникают длинные пробелы. Перенос слов может помочь избавится от

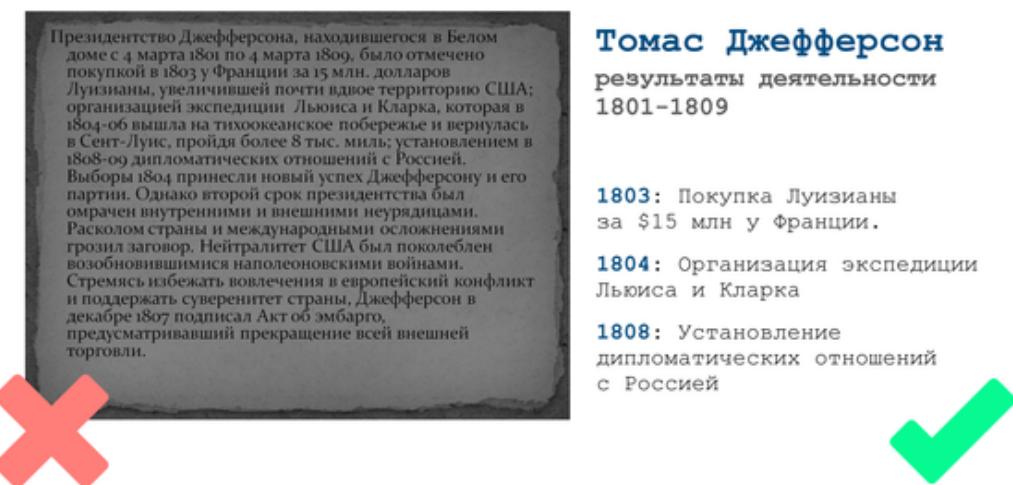


Рисунок 7.19. Плохо структурированное сообщение читать не хочется, оно кажется скучным. В этом примере меньшее количество информации хорошо оформленного текста полезнее, чем много неструктурированного текста, который прочитает мало людей



Пример текста, который вряд ли прочитает кто-то, кроме директора и пиарщика

Пример статьи, которая соберет десятки тысяч просмотров, если ее правильно распространять

Рисунок 7.20. Структура страницы, шрифтовая иерархия должна облегчать сканирующее чтение

Чем отличаются относительные и абсолютные риски

УЗНАТЬ ↓

Омикрон обладает меньшей природной вирулентностью почти для всех. Но не для детей

Первые оценки вирулентности для невакцинированных и неболевших, о которых уже писала «Медуза», говорили о том, что омикрон, видимо, реже вызывает тяжелые заболевания, чем дельта. Для людей, которые никогда не сталкивались с SARS-CoV-2 и не были привиты, риск госпитализации оценивался примерно на 25–30% меньше. Новое исследование группы Фергюсона даже улучшило эту оценку — в среднем по популяции относительный риск попасть в больницу при омикроне по сравнению с дельтой уменьшился на 70% (но речь здесь только о тех, у кого нет никакого иммунитета).

Пример:

Допустим, в некоем абстрактном городе, где еще нет вакцинированных, происходит распространение варианта дельта.

Рисунок 7.21. Фрагмент страницы с сайта meduza.io, где приведена большая статья о новом штамме COVID-19. Помимо заголовка, выделенной шрифтом без засечек аннотации, в статье выделены подзаголовки, некоторые абзацы скрыты, отдельно отмечены примеры и главные идеи. Такая структура страницы облегчает сканирующее чтение.

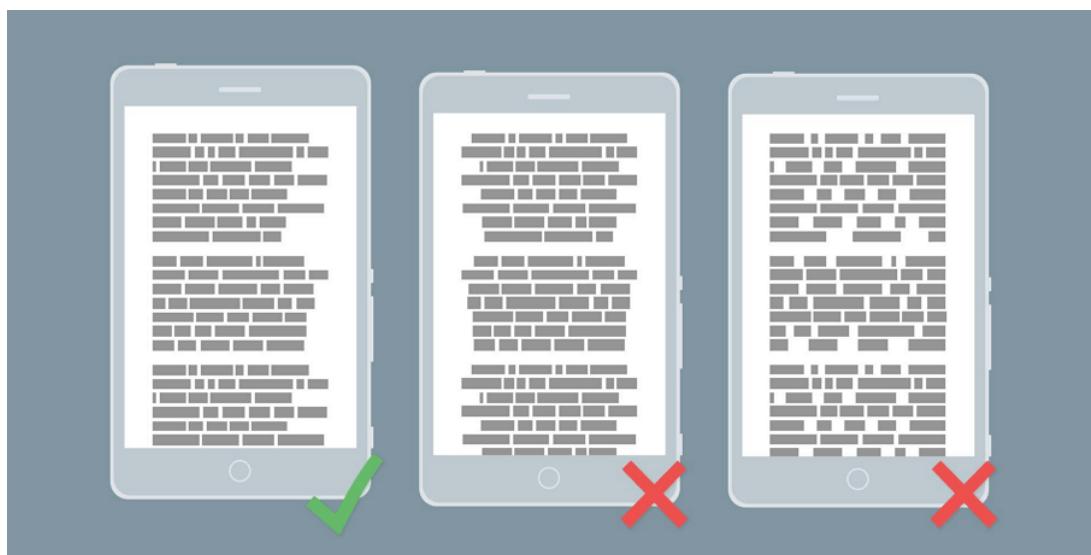


Рисунок 7.22. На сайтах уместнее использовать выравнивание больших блоков текста по левому краю, избегая непропорциональных пробелов.

[Печатный текст](#) | [править](#) | [править код](#)

Традиционные исследования длины строки, ограничивавшиеся печатным текстом, сообщали о разных результатах, но обычно для печатного текста принята длина строки от 45 до 75 [знаков на строку](#)^[6] (англ. *characters per line*, *cpl*), хотя идеальным вариантом является 66 *cpl*, включая символы и пробелы^[11]. В книжном наборе длина строки обычно равна 30 размерам набора, но считается допустимым набор в пределах 20–40 раз (то есть, $30 \times 10\text{-пунктный набор} = 300\text{-пунктная строка}$)^[11]. Ранние исследования считали длину строки в 59–97 [мм](#) (около 57 *cpl*) оптимальной для 10-пунктного набора^[2]. Для печатных работ с несколькими колонками часто лучше 40–50 *cpl*^[1]. Для текста на английском языке с [выключкой по формату](#) минимальное число знаков на строку – 40, длина строки менее 38–40 символов часто приводит к появлению проблем (или «коридоров») или слишком большого числа переносов слов в блоке текста^[1]. Длинные строки (85–90 *cpl*) могут быть допустимы для несвязанного текста, например, в [библиографиях](#) или [сносках](#), но для непрерывного текста строки в более чем 80 символов могут быть слишком длинными. Короткий текст, например, [равные маргиналии](#), может быть и длиной в 12–15 знаков на строку^[11]. Исследования показали, что более краткие строки часто предпочитаются длинным, вероятно, потому что участникам более привычен такой формат^[3].

[Электронный текст](#) | [править](#) | [править код](#)

Чтение с экрана создаёт дополнительные проблемы, делая использование традиционных исследований длины текста для электронных форматов проблематичным^[4]. В отличие от печатного текста, дизайн для цифровых носителей должен учитывать такие факторы, как блики, мерцание и прокрутка или перелистывание^[5].

Исследования читаемости цифрового текста показали, что, как и для печатного текста, длина строки может влиять на скорость чтения. Если строки слишком длинные, читателям трудно вернуться к началу следующей строки ([«скакада»](#)), в то время как если они слишком короткие, им требуется больше прокрутки или листания^[6]. Исследователи полагают, что длинные строки лучше подходят для быстрого сканирования, а короткие строки – для точности чтения^[3]. Одно из предложений для наилучшего компромисса между скоростью чтения и пониманием – использовать около 55 знаков на строку^[6]. В то же время были и исследования, указывавшие, что цифровой текст длиной в 100 знаков может читаться быстрее, чем строки в 25 знаков, сохранив тот же уровень понимания текста^[4].

Субъективные факторы также играют роль в выборе длины строки для цифрового текста. Одно исследование показало, что число знаков на строку имело незначительное влияние на читаемость, включая факторы скорости и понимания текста, однако при запросе о своих предпочтениях 60 % участников исследования отдали предпочтение самым коротким (35 *cpl*) или самым длинным (95 *cpl*) строкам, использованным в исследовании. В то же время 100 % участников выбрали один из этих вариантов как наименее предпочтительный^[7].

Рисунок 7.23. Длина строки в статьях на wikipedia.org около 200 символов. Дойдя до конца строки при чтении сложнее вернутся к началу следующий.

длинных пробелов. В некоторых программах для вёрстки текстов возможна тонкая подстройка параметров выравнивания текста: задание ограничений на увеличение пробела, трекинга и размеров букв на 1–3%. Такие изменения практически незаметны, но могут помочь добиться ровной правой границы текста.

Считается, что оптимальная длина строки 40–70 знаков. При этом чем длиннее строка, тем больше должен быть интерлинияж – вертикальное расстояние между соседними строками. При больших объёмах текста у пользователя может рябить в глазах. В этом случае помогает снижение контрастности между текстом и фоном. Чаще всего можно сменить чёрный цвет текста на тёмно-серый.

7.3 Текст и синтаксис интерфейса

Как было отмечено, значительную часть информации человек получает в виде текста. Взаимодействия с пользователем практическим редко обходится без текста.

Хорошая типографика не исправит ситуацию, если текст написан плохо: его трудно понять, он слишком длинный, там много “воды”, сложных речевых оборотов и незнакомых пользователю терминов. Поэтому важно ещё и уметь эффективно доносить информацию через тексты, надписи и пояснения.



В чём важность типографики?

Ян Чихольд и Роберт Бринхёртс не могли прийти к единому мнению в определении типографики, но они были полностью сходятся в том что является её целями. Они лишь использовали разные слова: "Суть новой типографики — ясность" — выделено жирным шрифтом в книге Чихольда. "Новая типография отличается от старой тем фактом, что ее первой задачей является разработка ее видимой формы из функций текста ... форма должна быть создана из функции" — уточняет Чихольд.

Бринхерст был более конкретным. Типография должна приглашать читателя в текст, раскрывать смысл и настроение текста, уточнять его структуру и порядок и связывать текст с другими существующими элементами.



Большинство людей думает, что типография касается шрифтов. Большинство дизайнеров считают, что типография — это шрифты.

Типография больше, чем это, она выражает язык через шрифт.
Размещение, состав, выбор шрифта.

Марк Боултон

Типография — это не только шрифты. Речь идет не о размере шрифта, краях, интервалах, масштабах или цвете. Типография посвящена объективной организации информации. Речь идет о информации и о том, как мы передаем эту информацию читателю. Размер, цвета и шрифты — это просто инструменты, которые помогают нам превращать эту информацию в правильную форму. И, как мы увидим, все эти инструменты должны работать вместе в целом. Нет отдельных элементов.

Рисунок 7.24. Длина строки в статьях на medium.com около 60 символов.

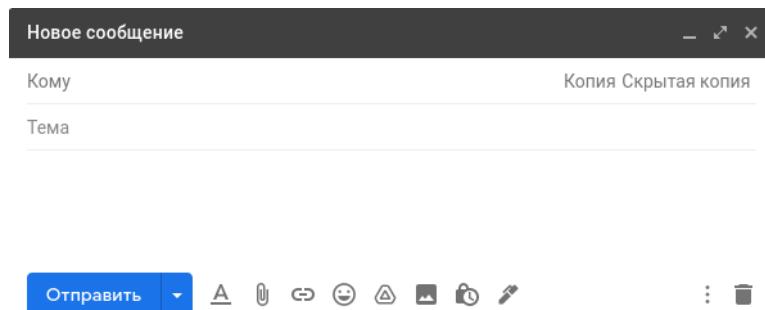


Рисунок 7.25. Форма отправки сообщения в Gmail. Подлежащее – заголовок (существительное), сказуемое – кнопка отправить (глагол).

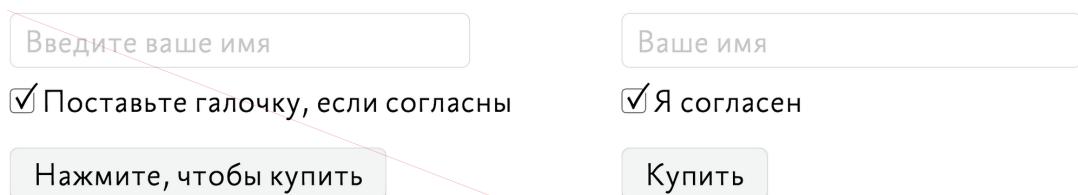


Рисунок 7.26. Синтаксис интерфейса составляют не только слова, но и элементы управления, поля ввода, флажки, переключатели и другие элементы. Поэтому не нужно добавлять к кнопке слово “нажмите”, к полю ввода слово “введите” и т.д.

Многие диалоговые окна и формы можно рассматривать как предложения, которые образуют связный текст или предложение (рис. 7.25). Такой взгляд подражает привычной для человека подаче информации, с той лишь разницей, что из элементов интерфейса не всегда можно сложить текст механическим соединением. Элементы интерфейса имеют те же синтаксические отношения, что и слова в предложениях. Одни элементы интерфейса можно представлять главными членами предложениями – подлежащим¹ и сказуемом².

Кнопки и другие элементы интерфейса, отвечающие за совершение действий, стоит подписывать глаголами. Это особенно полезно в диалоговых окнах, где пользователь может понять смысл сообщения из подписей к кнопкам (рис. 7.27).

Кнопки и элементы меню могут ещё означать изменение статуса. Например “отметить письмо как Важное” или “отметить

¹Подлежащее – главный член предложения, который обозначает предмет, действие которого выражается сказуемым

²Сказуемое – главный член предложения, которым бывает глагол, связанный с подлежащим и отвечающий на вопросы: что делает предмет? что с ним происходит? каков он? и др; обозначает действие или состояние.

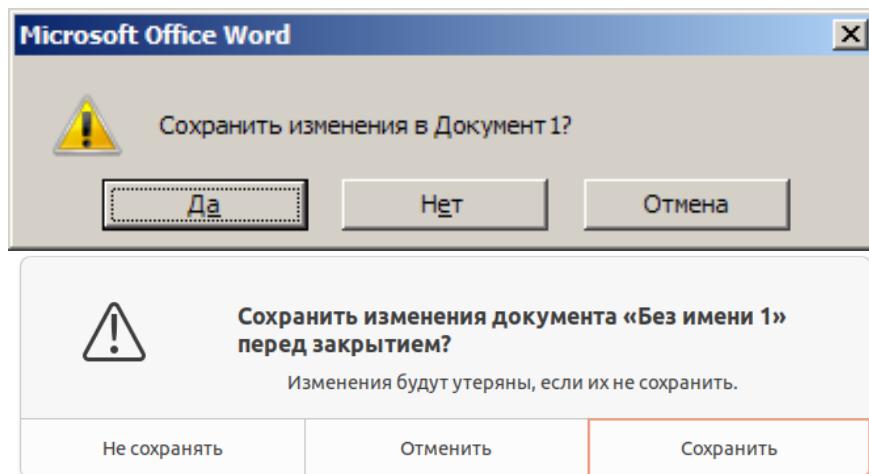


Рисунок 7.27. В старых версиях Microsoft Word из названий кнопок сразу нельзя понять о чем идёт речь. В современных текстовых редакторах все кнопки подписаны глаголами, можно догадаться о чем идёт речь только прочитав названия кнопок.

письмо как Спам“. В таких случаях уместно выносить общий текст (“отметить письмо как“) за скобки и оставить на кнопках только статус: “Важное“ и “Спам“.

Большая часть действий пользователя – это применение некоторого *действия* к некоторому *объекту*. Например, *изменения шрифта для абзаца текста* в текстовом редакторе. В интерфейсе могут использоваться две последовательности: сначала выбор глагола (изменить шрифт), а затем выделение существительного (абзац); сначала существительное, а потом глагол.

Эксперименты [3] показывают, что в большинстве случаев модель существительное → глагол предпочтительнее. Это уменьшает количество ошибок. Последовательность глагол → существительное устанавливает *режим*. Это повышает скорость: пользователю не требуется переключать своё внимание с содержания (которое и вызвало необходимость выполнения операции) к самой команде и затем опять к содержанию. Помогает отменять действия: при использовании модели глагол → существительное должна быть предусмотрена возможность отмены команды. Если пользователь назначает команду и затем решает

изменить её, следует учитывать, что он находится в этот момент в режиме, и система ожидает, что будет сделано выделение текста для применения уже назначеннной команды.

Особено важно донести до пользователя информацию, когда он или программа совершили ошибку. Сообщения об ошибках должны соответствовать трём критериям (рис. 7.28):

- быть понятным
- быть лаконичным
- быть полезным.

Приведённые в первой части примеры сообщений об ошибках (рис 1.4) соответствуют только второму критерию.

Помимо интерфейсного текста, разработчик пишет справку, документацию, инструкции и иногда сопроводительный текст для сайта приложения или страницы в Google Play или App Store. Здесь требования лаконичности и ясности не менее важны.

Необходимость сжато и ясно передавать информацию текстом появилась задолго до пользовательских интерфейсов. Яркий пример сокращения текста демонстрирует Ольминский М. С.³ на примере новостной заметки о демонстрации в Твери. Текст заканчивался словами: “Явившаяся на место происшествия местная полиция арестовала восемь человек демонстрантов”. Ольминский критикует: “Местная” – разве в Твери могла явиться полиция не местная, а казанская?. “Явившаяся на место происшествия” – разве могла она арестовать, не явившись? “Полиция” – кто же арестует, кроме полиции? “Человек демонстрантов” – конечно, не коров и не прохожих. Остаётся для печати: “Арестовано восемь”.

Упростить редактуру могут сервисы:

- orfogrammka.ru – проверка орфографии, пунктуации, сти-

³Ольминский Михаил Степанович – революционер, публицист, историк, литературный критик, литературовед и историк литературы

Original	Failure An authentication error has occurred OK
Clear	Sign-in error You entered an incorrect password OK
Clear, Concise	Wrong password OK
Clear, Concise, Useful	Wrong password TRY AGAIN RECOVER PASSWORD

Рисунок 7.28. 3 стадии улучшения сообщения (блок Original): Сбой. Произошла ошибка аутентификации. 1. Сделать сообщение понятным (clear): Ошибка входа. Вы ввели неправильный пароль. 2. Сделать лаконичным (concise): Неправильный пароль. 3. Сделать полезным (useful), добавить кнопки “попробовать ещё раз” и “восстановить пароль”.

листики и элементов типографики;

- glvrd.ru – Главред – оценка текста с точки зрения информационного стиля [6];
- hemingwayapp.com – оценка читаемости и понятности текстов на английском языке.

Вопросы

1. Что такое типографика?
2. Что такое удобочитаемость и различимость? Чем они отличаются?
3. Приведите классификации шрифтов?
4. Что такое антиква и готеск? Брусковый шрифт?
5. Чем отличается моноширинный шрифт от пропорционального?



404. That's an error.

The requested URL /errorpage was not found on this server. That's all we know.

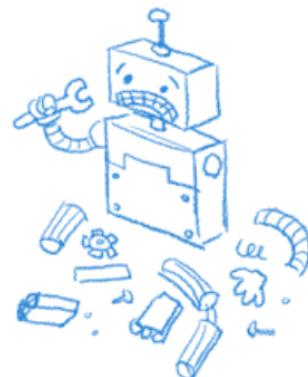


Рисунок 7.29. Гугл ничего не предлагает пользователю полезных действий на странице с ошибкой 404.

Яндекс



Найти

Ошибка 404. Нет такой страницы

Если вы считаете, что страницы нет по нашей вине, [напишите нам](#).

- Эфир — фильмы 1960-х годов
- Маркет — смартфоны Xiaomi
- Музыка — для полезной изоляции
- Игры — прямо в браузере
- Практикум — учёба с наставником
- Недвижимость — дизайнерский ремонт
- Расписания — поезда и электрички

Рисунок 7.30. Яндекс на странице с ошибкой 404 показывает строку поиска и предлагает обратную связь

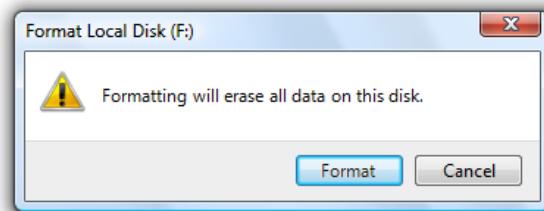
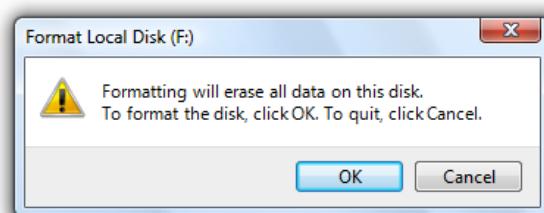


Рисунок 7.31. Глаголы предпочтительнее существительных там, где идёт речь о совершении действия

6. Где полезно использовать моноширинный шрифт?
7. Что такое акцидентный шрифт?
8. Какие шрифты называют нейтральными, а какие характерными?
9. Что такое шрифт и гарнитура?
10. Какие бывают начертания?
11. Что такое переменный шрифт?
12. Что такое кернинг?
13. Опишите рекомендации по выбору шрифтов.
14. Что такое шрифтовая иерархия? Как её можно показывать?
15. Что такое сканирующее чтение?

Литература

- Бирман, И. Пользовательский интерфейс / И. Бирман. — : Дизайн-бюро Артёма Горбунова, 2017. — 419 с.
- Ильяхов, М. Пиши, сокращай: Как создавать сильные тексты / М. Ильяхов, Л. Сарычева, . — : Альпина Паблишер, 2021. — 440 с
- Горбунов А. С. Типографика и вёрстка. — М.: Изд-во Бюро Горбунова, 2015. - 175 с.

Заключение

В пособии рассмотрены элементы системы "человек – машина". Дано описание важным с точки взаимодействия с пользовательским интерфейсом (ПИ) характеристикам человека: вниманию, памяти, способности формировать привычки. Описаны особенности восприятия визуальной информации человеком.

Приведена классификаций ПИ; парадигмы, на основе которых строится понимание интерфейса пользователем. Рассмотрена методология проектирования ПИ продукта с точки зрения ПИ в частности и пользовательского опыта (UX) вообще.

Рассмотрена основная качественная характеристика ПИ – юзабилити, некоторые способы юзабилити-тестирования, количественной оценки интерфейсов. Метод GOMS.

Наконец приведены общие сведения о типографике и написанию текста в ПИ.

Все замечания по содержанию учебного издания можно направлять на почтовый ящик vetrovsv@outlook.com. Автор будет признателен за любые конструктивные предложения.

Литература

- [1] Норман Д., Дизайн привычных вещей, обновлённое и дополненное издание — : Манн, Иванов и Фербер, 2021. — 384 с.
- [2] Канеман Д. Думай медленно... решай быстро — : АСТ, 2013. — 710 с.
- [3] Раскин Д., Интерфейс: новые направления в проектировании компьютерных систем. – Пер. с англ. – СПб: Символ-Плюс, 2007. – 272 с.
- [4] Д. Канеман. Внимание и усилие / под ред. А.Н. Гусева. — Москва: Смысл, 2006. — 287 с.
- [5] Мильчин А.Э., Методика редактирования теста. Изд. 3-е, перераб. и доп. - М.: Логос, 2005. - 524 с.
- [6] Ильяхов, М. Пиши, сокращай: Как создавать сильные тексты / М. Ильяхов, Л. Сарычева, . — : Альпина Паблишер, 2021. — 440 с
- [7] Бирман, И. Пользовательский интерфейс / И. Бирман. — : Дизайн-büро Артёма Горбунова, 2017. — 419 с.
- [8] Пользовательский интерфейс: прицеливание. — Текст : электронный // Дизайн-büро Артёма Горбунова : [сайт]. — URL: <https://bureau.ru/books/ui/demo/4> (дата обращения: 20.08.2021).
- [9] Гарретт Дж. Вебдизайн: книга Джессса Гарретта. Элементы опыта взаимодействия». – Пер. с англ. – СПб.: СимволПлюс, 2008. – 192 с.: ил

- [10] Алан Купер, Рейманн Роберт М., Основы проектирования взаимодействия. – Пер. с англ. – СПб.: Символ-Плюс, 2018, 720 с.
- [11] Alan Cooper, About Face: The Essentials of Interaction Design, Fourth Edition, 2014, 722 p.
- [12] Купер, А. Психбольница в руках пациентов. Алан Купер об интерфейсах / А. Купер. — : Питер, 2018. — 384 с.
- [13] Буч Г., Рамбо Д., Якобсон И. Язык UML. Руководство пользователя. 2-е изд.: Пер. с англ. Мухин Н. – М.: ДМК Пресс, 2006. – 496 с.: ил.
- [14] Выготский Л.С., Психология. – Москва : Эксмо-Пресс : Апрель Пресс, 2000. – 1008 с.
- [15] SO/IEC/IEEE 24765-2010 – Systems and software engineering Vocabulary
- [16] Медведев, В.Ю. Роль дизайна в формировании культуры / В.Ю. Медведев. – СПб.: СПГУТД, 2004. – 108 с.
- [17] Горбунов А. С. Типографика и вёрстка. – М.: Изд-во Бюро Горбунова, 2015. - 175 с.
- [18] Google Fonts. — Текст : электронный // : [сайт]. — URL: <https://fonts.google.com/> (дата обращения: 15.09.2021).
- [19] Variable Fonts. — Текст : электронный // : [сайт]. — URL: <https://v-fonts.com/> (дата обращения: 20.02.2022).
- [20] Edward R. Tufte, The Visual Display of Quantitative Information, Graphics Press, 2001. 200 p.
- [21] Edward R. Tufte Envisioning Information, Graphics Press, 1990. 126 p.
- [22] Edward R. Tufte, Visual Explanations: Images and Quantities, Evidence and Narrative Graphics Press, 1997, 156 p.
- [23] Edward R. Tufte, Beautiful Evidence, Graphics Press, 2006, 213 p.

- [24] Стив Круг. Не заставляйте меня думать; - 3-е издание. – Москва: Издательство ‚; 2018. - 256 с.
- [25] Хьюбел, Д. Глаз, мозг, зрение / Д. Хьюбел. — : Мир, 2003. — 240 с. — Текст : непосредственный.
- [26] Унгер, Р. UX-дизайн. Практическое руководство по проектированию опыта взаимодействия / Р. Унгер, К. Чендлер. — : Символ-Плюс, 2011. — 336 с. — Текст : непосредственный.
- [27] Голомбински, К. Добавь воздуха! Основы визуального дизайна для графики, веба и мультимедиа / К. Голомбински, Р. Хаген. — : Питер, 2013. — 272 с. — Текст : непосредственный.
- [28] Орфограммка. — Текст : электронный // : [сайт]. — URL: <https://orfogrammka.ru/> (дата обращения: 23.01.2022).
- [29] Главред. — Текст : электронный // : [сайт]. — URL: <https://glvrd.ru/> (дата обращения: 23.01.2022).
- [30] Paradigm. — Текст : электронный // Дизайн-система Mail.ru Paradigm : [сайт]. — URL: <https://paradigm.mail.ru/> (дата обращения: 23.01.2022).
- [31] Human Interface Guidelines. — Текст : электронный // developer.apple.com : [сайт]. — URL: <https://developer.apple.com/design/human-interface-guidelines/> (дата обращения: 21.07.2021).
- [32] Material Design Guidelines. — Текст : электронный // material.io : [сайт]. — URL: <https://material.io/design/guidelines-overview> (дата обращения: 21.07.2021).
- [33] Design and code Windows apps. — Текст : электронный // microsoft.com : [сайт]. — URL: <https://docs.microsoft.com/en-us/windows/apps/design/> (дата обращения: 21.02.2022).
- [34] Fluent Design System. — Текст : электронный // microsoft.com : [сайт]. — URL:

<https://www.microsoft.com/design/fluent/> (дата обращения: 21.07.2021).

[35] GNOME Human Interface Guidelines. — Текст : электронный // gnome.org : [сайт]. — URL: <https://developer.gnome.org/hig/> (дата обращения: 21.07.2021).

[36] KDE Human Interface Guidelines. — Текст : электронный // kde.org : [сайт]. — URL: <https://develop.kde.org/hig/> (дата обращения: 21.07.2021).

[37] typographic system with modular scale vertical rhythm. — Текст : электронный // Gridlover : [сайт]. — URL: <https://www.gridlover.net/try> (дата обращения: 11.10.2021).

[38] Type Scale - A Visual Type Scale. — Текст : электронный // type-scale.com : [сайт]. — URL: <https://type-scale.com/> (дата обращения: 11.10.2021).

[39] Зараменских, Е. П. Управление жизненным циклом информационных систем : учебник и практикум для вузов / Е. П. Зараменских. — 2-е изд. — Москва : Издательство Юрайт, 2022. — 497 с. — (Высшее образование). — ISBN 978-5-534-14023-1. — Текст : электронный // Образовательная платформа Юрайт [сайт]. — URL: <https://urait.ru/bcode/489983> (дата обращения: 23.01.2022).

[40] Одегов, Ю. Г. Эргономика : учебник и практикум для академического бакалавриата / Ю. Г. Одегов, М. Н. Кулапов, В. Н. Сидорова. — Москва : Издательство Юрайт, 2018. — 157 с. — (Бакалавр. Академический курс). — ISBN 978-5-9916-8258-9. — Текст : электронный // Образовательная платформа Юрайт [сайт]. — URL: <https://urait.ru/bcode/413951> (дата обращения: 23.01.2022).

[41] Климов Е.А. Инженерная психология и эргономика : учебник для академического бакалавриата / Е. А. Климов [и др.] ; под редакцией Е. А. Климова, О. Г. Носковой, Г. Н. Солнцевой. — Москва : Издательство Юрайт, 2018. — 178 с. — (Ба-

калавр. Академический курс. Модуль). — ISBN 978-5-534-00906-4. — Текст : электронный // Образовательная платформа Юрайт [сайт]. — URL: <https://urait.ru/bcode/415260> (дата обращения: 23.02.2022).

- [42] Магазанник В. Д. Человеко-компьютерное взаимодействие : учеб. пособие для вузов / Магазанник В. Д. - 2-е изд., доп. и перераб. - М. : Университетская книга, 2016. - 406 с.
- [43] Мандел, Т. Разработка пользовательского интерфейса / Т. Мандел. — Москва : ДМК Пресс, 2007. — 418 с. — ISBN 5-94074-069-3. — Текст : электронный // Лань : электронно-библиотечная система. — URL: <https://e.lanbook.com/book/1227>
- [44] Сакулин, С. А. Основы интернет-технологий: HTML, CSS, JavaScript, XML : учебное пособие / С. А. Сакулин. — Москва : МГТУ им. Н.Э. Баумана, 2017. — 112 с. — ISBN 978-5-7038-4724-4. — Текст : электронный // Лань : электронно-библиотечная система. — URL: <https://e.lanbook.com/book/103525>
- [45] Винокуров, И. В. Использование библиотек классов Swing и MFC для разработки графического интерфейса пользователя : учебное пособие / И. В. Винокуров. — 2-е изд., стер. — Москва : МГТУ им. Н.Э. Баумана, 2011. — 396 с. — ISBN 978-5-7038-3560-9. — Текст : электронный // Лань : электронно-библиотечная система. — URL: <https://e.lanbook.com/book/106519>
- [46] Алфимцев А. Н. Разработка пользовательского интерфейса [Электрон. ресурс] : метод. указания к лаб. раб. по дисциплине "Протоколы и интерфейсы информационных систем" / Алфимцев А. Н. ; МГТУ им. Н. Э. Баумана. - М. : МГТУ им. Н. Э. Баумана, 2010.
- [47] Алфимцев А.Н. 25 упражнений по юзабилити. Saarbrücken, Deutschland, LAP Lambert Academic Publishing. 2014. 108 с.

- [48] Сакулин С. А., Алфимцев А. Н. Основы HTML [Электрон. ресурс] : метод. указ. к лабораторно-вычислительной практике по дисциплине "Интернет-технологии"/ Сакулин С. А., Алфимцев А. Н. ; МГТУ им. Н. Э. Баумана, Фак. "Информатика и системы управления". - М. : Изд-во МГТУ им. Н. Э. Баумана, 2013. - 1 CD-ROM. - ФГУП "Информрегистр" №0321302426.
- [49] Алфимцев А. Н., Девятков В. В. Интеллектуальные мультимодальные интерфейсы / Алфимцев А. Н., Девятков В. В. - Калуга : Полиграф-Информ, 2011. - 325 с. : ил. - Библиогр.: с. 285-322.
- [50] Брокшmidt, К. Пользовательский интерфейс приложений для Windows 8, созданных с использованием HTML, CSS и JavaScript [Электронный ресурс] : учебное пособие / К. Брокшmidt. — Электрон. дан. — Москва : , 2016. — 395 с. — Режим доступа: <https://e.lanbook.com/book/100369>.
- [51] Попов А. А. Эргономика пользовательских интерфейсов в информационных системах : учеб. пособие / Попов А. А. - М. : РУСАЙНС, 2017. - 311 с. : ил. - Библиогр.: с. 304-311.
- [52] Винокуров, И.В. Использование библиотек классов Swing и MFC для разработки графического интерфейса пользователя [Электронный ресурс] : учебное пособие / И.В. Винокуров. — Электрон. дан. — Москва: МГТУ им. Н.Э. Баумана, 2011. — 396 с. — Режим доступа: <https://e.lanbook.com/book/106519> .
- [53] ГОСТ Р МЭК 60447-2000 Интерфейс человеко-машинный. Принципы приведения в действие.
- [54] ГОСТ Р МЭК 60073-2000 Интерфейс человекомашинный. Маркировка и обозначение органов управления и контрольных устройств. Правила кодирования информации.
- [55] Squire, L. R., Knowlton, B. J. The medial temporal lobe, the hippocampus, and the memory systems of the brain. In M.

Gazaniga (Ed.), The new cognitive neurosciences (2nd ed., pp. 765–780). Cambridge, MA: MIT Press., 2000

- [56] George A. Miller. The Magical Number Seven, Plus or Minus Two. // The Psychological Review, 1956, vol. 63, pp. 81—97.
- [57] The precipitous rise of Figma and fall of InVision. — Текст : электронный // : [сайт]. — URL: <https://uxdesign.cc/the-precipitous-rise-of-figma-and-fall-of-invision-435f07e8d1b6> (дата обращения: 27.07.2021).
- [58] The magical number seven, plus or minus two: Not relevant for design [Электронный ресурс]. – Режим доступа: https://www.edwardtufte.com/bboard/q-and-a-fetch-msg?msg_id=0000U6 (дата обращения: 7.11.2021)
- [59] Schneider W., Shiffrin R.M. Controlled and automatic human information processing: Detection, search and attention // Psychol. Review. 1977. Vol. 84. N 1. P. 1—66.
- [60] Don, Norman; Jakob, Nielsen. "The Definition of User Experience (UX)". Nielsen Norman Group. Retrieved 2 March 2021.
- [61] Хабр: Принципы гештальта в дизайне пользовательского интерфейса [Электронный ресурс]. – Режим доступа: <https://habr.com/ru/company/cloud4y/blog/347444/>. – Дата доступа: 13.09.2021.
- [62] Visualizing Fittss law. — Текст : электронный // Particletree : [сайт]. — URL: <http://particletree.com/features/visualizing-fittss-law/> (дата обращения: 20.08.2021).
- [63] MacKenzie, I. S.. Fitts' law. In K. L. Norman J. Kirakowski (Eds.), Handbook of human-computer interaction, 2018, pp. 349-370.
- [64] Carroll, John Millar and Mary Beth Rosson. Paradox of the active user. 1987.

[65] Applying Writing Guidelines to Web Pages. — Текст : электронный // Nielsen Norman Group : [сайт]. — URL: <https://www.nngroup.com/articles/applying-writing-guidelines-web-pages/> (дата обращения: 21.02.2022).

[66] Csikszentmihalyi, Mihaly. Flow: The Psychology of Optimal Experience, 1990, 336 р.

[67] Т. Демарко, Т. Листер «Человеческий фактор: успешные проекты и команды». — Пер. с англ. — СПб: СимволПлюс, 2005. 256 с.

Учебное издание

Человеко-машинное взаимодействие

Учебное пособие набрано и отвёрстано в системе TeX

Печатается с оригинал-макета автора при участии
издательства

Подписано в печать **.*.2022 . Формат ** × ** '/**. .

• •

... **. . . **. № *****.

*** . -28 .).????

<< >> 672039, . , . -, 30