Джеф Раскин известен как специалист по компьютерным интерфейсам и инициатор проекта Macintosh или Mac — персональных компьютеров Apple. Свой опыт он обобщает в книге «The Humane Interface», которая была издана в 2000 году.

По мнению Раскина, интерфейс считается человекоориентированным, если он отвечает нуждам человека и учитывает его слабости. В книге он говорит, как разрабатывать интерфейсы с учетом принципов когнетики и решить проблемы взаимодействия человека с компьютером.

Здесь можно посмотреть весь текст книги <http://raskin-interface.narod.ru/interface/index.htm>

https://miro.medium.com/max/60/0*zH4xr9VD6Ai-a1sQ.jpg?q=20



Интерфейс — это то, как вы решаете задачу с помощью продукта. Т.е. это совершаемые вами действия и то, что вы получаете в ответ.

**Основные идеи**

**Простое должно оставаться простым**

Установить время на механических часах гораздо проще, чем на электронных. Поэтому их интерфейс лучше.

*Какой сложной бы не была система, простые задачи должны оставаться простыми.*

**Задача — проектирование — разработка**

Пользователи не задумываются над тем, как устроена машина, пока она справляется со своими задачами. Им не важен процессор и вся техническая начинка, имеет значение лишь удобство и результат. Но видят они только интерфейс — значит, для них это конечный продукт.

На первом этапе разработки нужно определить, что именно  
должен сделать пользователь для получения результата и как система должна отвечать на каждое его действие. В ходе работы определение задачи скорее всего будет меняться.

*Определив задачу, для которой предназначен продукт, сначала спроектируйте интерфейс, а потом приступайте к его реализации.*

**Человек концентрируется только на чем-то одном**

Одно из самых важных ментальных ограничений человека — у нас только один локус внимания. Это значит, что из всех объектов или явлений окружающего мира, которые вы воспринимаете с помощью своих чувств или воображения, в каждый момент времени вы можете сконцентрироваться только на одном.

Пользователь, поглощенный своей задачей, не замечает уведомления, подсказки и другие попытки системы пообщаться с ним.

Единственное, чем мы можем помочь — это сделать так, чтобы пользователь не мог совершить ошибок или имел возможность сразу отменить результаты своего действия.

*Хороший интерфейс работает независимо от степени поглощенности пользователя. Предупреждения о потенциальных ошибках бесполезны.*

**Нужно учитывать привычки**

При постоянном использовании интерфейса у вас формируются определенные привычки. Любая привычка означает отказ от внимания к деталям. В этом смысле задача дизайнеров заключается в том, чтобы создавать интерфейсы, которые не позволяют привычкам вызывать проблемы у пользователей.

Многие компьютеры перед неизбежным действием задают вопрос: «Вы уверены?» После этого вам требуется ввести либо «Да», либо «Нет» в качестве ответа. Пользователь привыкает отвечать «Да» на любую команду. Из-за постоянного повторения это становится привычным действием и, вместо того чтобы остаться отдельной ментальной операцией, превращается в часть действия по удалению файлов.

*Любой запрос о подтверждении, требующий установленного ответа, вскоре становится бесполезным.*

**Обратной связи недостаточно**

Разные режимы ведут к ошибкам. Средств обратной связи и индикаторов текущего положения системы недостаточно, чтобы обеспечить устранение ошибок, вызванных режимами. Потому что это не находится в локусе внимания.

Лучше избегать режимов. Когда появятся привычки в пользовании интерфейсов, не нужно будет задумываться или планировать действие, поэтому внимание будет сосредоточено только на содержании работы.

**Не всегда нужны лишние настройки**

Большинство пользователей просто хотят выполнить свою работу. Их не волнует, какой шрифт принят по умолчанию и т.д.

Добавляя в программу настройки, мы обременяем пользователя новой задачей: скажем, помимо электронных таблиц, приходится изучать еще и опции по их настройке.

*Время, которое тратится на изучение и выполнение пользовательских настроек, — потерянное с точки зрения текущей задачи.*

**Пользователь слепой**

По отношению к тому, что находится вне нашего локуса внимания, мы все в самом прямом смысле слепые. Поэтому элементы интерфейса называют привычными, если ими может пользоваться «слепой» пользователь.

**Нажимать кнопку или нажимать и удерживать?**

Пользователь должен уделять время содержанию и работе с ним, а не управлению системой. Поэтому квазирежимы (когда нужно удерживать клавишу) должны использоваться для управленческих функций, тогда как для создания содержания должны применяться операции без задействования квазирежимов.

**Правильный порядок действий: сначала выбирать объект, потом действие с ним**

**Модель «существительное-глагол»:** сначала выбрать объект (выделить абзац), а потом указать действие (выбрать шрифт).

**Модель «глагол- существительное»:** сначала выбрать действие (изменить шрифт), потом — объект (абзац).

Подход «существительное-глагол» лучше. Преимущества:

* Уменьшение количества ошибок — команды выполняются сразу, пока еще находятся в локусе внимания пользователя.
* Скорость — не нужно переключать свое внимание с содержания (которое и вызвало необходимость выполнения операции) к самой команде и затем опять к содержанию, чтобы можно было выделить необходимый участок текста.
* Простота и обратимость — чтобы выбрать другой текст, не требуется нажимать кнопку отката или применять какой-либо другой способ отмены действия.

Применение методов «глагол-существительное» должно ограничиваться только выбором из палитр, если они предназначены для непосредственного использования.

**Делать объекты состоятельными**

Элементы управления считаются состоятельными (affordance), когда по внешнему виду элемента понятно, какая у него функция. Если будет ручка — пользователи будут ее крутить, если кнопка — нажимать.

То, какую состоятельность будет иметь элемент интерфейса, зависит от опыта и знаний пользователя, а также от контекста, в котором он этот элемент встречает. Каждый видимый элемент должен быть состоятельным.

**Делать жесты монотонными**

Монотонность — это вторая сторона отсутствия модальности. В интерфейсе, который не имеет режимов, данный жест пользователя может иметь один и только один результат: жест g всегда приводит к действию a.

Перед разработчиками интерфейсов стоит задача по поиску монотонного решения, чтобы обеспечить такие преимущества: легкость изучения, простоту внедрения, минимум документации и небольшой размер расходов на обслуживание. Суть монотонности — для вызова одной и той же команды не должно использоваться много жестов.

**Дихотомия «новичок-эксперт» — миф**

Фаза профессионального владения интерфейсом характеризуется бессознательным использованием его возможностей. Поэтому менять интерфейс на другой при достижении какого-то уровня — раздражать пользователя.

Простота и ясность функций и их видимость важны и для новичков, и для экспертов. Причем переход от одного статуса к другому не должен требовать переучивания.

***Двойная «дискликсия»***

Интерфейсная техника, называемая «двойным кликом», т. е. двойное нажатие кнопки за короткий временной промежуток и без какого-либо значительного перемещения курсора между двумя нажатиями, имеет некоторые недостатки. То, каким образом двойной клик используется во многих сегодняшних интерфейсах, вынуждает пользователей запоминать не только то, по каким именно элементам можно дважды щелкать мышью, но и то, какой результат возникает в ответ на этой действие по отношению к разным классам элементов интерфейса.

**Уменьшать задержки или делать их понятными**

Если при использовании какого-то управляющего элемента на экране монитора в течение приблизительно 250 мс ничего не возникает, пользователь, скорее всего, может почувствовать беспокойство, решит сделать еще одну попытку или подумает, что система неисправна.

Если задержки неизбежны, в интерфейсе должна быть обратная связь, которая о них сообщает. Например, индикатор хода задачи (status bar), отражающий оставшееся время. Если неизвестно, сколько времени займет операция, так и скажите пользователю.

*Хороший интерфейс выдает сообщение, что информацию от пользователя программа приняла и правильно распознала.*

**Унифицировать команды**

Если сделать во всех приложениях одинаковые команды, можно устранить модальность.

*Разработка интерфейсов должна быть основана на идее, что любые объекты, которые выглядят одинаково, одинаковы.*

**Каталог элементарных действий**

Элементарные действия, выполняемые пользователем в различных комбинациях, порождают набор элементарных операций, которые применяются к содержанию и используются почти во всех интерфейсах. Перечислим, какие операции пользователи применяют к содержанию:

* *Указание.* Указывают на то или иное содержание.
* *Выделение.* Выделяют какое-то содержание.
* *Активизация.* С помощью «клика» активизируют содержание.
* *Модификация* или использование (с помощью команд):
* *Генерация.* Модификация из «пустого» в «непустое».
* *Удаление.* Модификация из «непустого» в «пустое».
* *Перемещение.* Вставка содержания в одно место и одновременное его удаление из другого.
* *Трансформация.* Преобразование в другой тип данных.
* *Копирование.* Содержание может быть отправлено или получено от внешнего устройства или скопировано в другую область внутри системы. Например, содержание распечатывают, отправляют по электронной почте, сохраняют на жестком диске, копируют в другой документ.

***Подсветка***(highlighting) означает, что с помощью каких-либо средств отображенному на экране объекту придается заметное отличие. Когда пользователь наводит курсор на какие-то объекты, они должны быть подсвечены.

***Указание***(indication) — это подсветка единичного объекта при перемещении курсора без каких-то других действий со стороны пользователя (например, без клика мыши). С помощью указания пользователь может в любой момент знать, на какие объекты он указывает с точки зрения системы.

***Выделение***(selecting) — это процесс, с помощью которого пользователь указывает, что один или несколько объектов имеют особый статус, который может быть воспринят системой. Как результат процесса получается выборка (selection).

Три недостатка выборок:

1. Команда для создания составных выборок невидимая.
2. Когда создаешь большую выборку, иногда случайно отпускаешь «Shift» и щелкаешь по следующему файлу, — вся выборка отменяется.
3. Механизм — «переключатель»: один и тот же жест отменяет и устанавливает выделение.

Решения:

1. Сделать экранную подсказку.
2. Установить специальную команду, с помощью которой текущая выборка определяется как объединение старого и текущего выделения. Благодаря такой команде пользователь сосредоточился на создании выборки, не заботясь о том, что было выбрано до этого.
3. Сделать разные команды для установки и удаления. Операторы *Отменить* и *Повторить* являются основополагающими, и их функция настолько важна, что в будущих системах для них должна быть предусмотрена специальная клавиша.

Когда процессом взаимодействия управляет человек, в фокусе обычно находится текущая выборка. Если же система отвечает на действие пользователя или внешней системы, в фокусе обычно находится результат действия.

**Подстраивать команды под естественный язык**

Синтаксис, который мы выбираем для написания команд, не должен исключать пробелы или символы новой строки. Любые ограничения, накладываемые на набор допустимых символов, используемых для набора команд, в будущем приведут к проблемам.

*Следует сделать так, чтобы машина подстраивалась под нас, а не подстраивать обычаи естественной речи под то, что проще решить с точки зрения программирования.*

**Не подвергать работу пользователя риску**

Человекоориентированный интерфейс никогда не подвергает работу пользователя риску => ввод с клавиатуры не замещает выделенный текст и не отменяет выделение текущей выборки. Случайная потеря хотя бы одного символа может означать потерю части телефонного номера, которая не может быть восстановлена.

*Удаление текста должно проходить явным образом по желанию пользователя и не быть побочным эффектом другого действия.*

Побочный эффект — это следствие применения команды, при котором изменяется содержание или события, которые не находятся в локусе вашего внимания. В только что рассмотренном случае в локусе вашего внимания находится текст, который вставляется, а побочным эффектом является удаление. Устранение побочных эффектов должно быть одной из целей для разработчика человекоориентированпого интерфейса.

Раскин советует отказаться от функции перетаскивания. Меньшее число ошибок и проблем будет хорошей компенсацией для пользователей, которые привыкли перетаскивать текст.

**Отказаться от имен файлов и иерархической структуры**

Необходимость давать имена файлам увеличивает ментальную нагрузку на пользователя. Это одна из главных бедствий, которым страдают современные компьютерные системы. Между именем файла и самим файлом не должно быть различия.

Человеческий мозг способен более эффективным образом использовать быструю, полнотекстовую поисковую систему, поэтому любое слово или фраза из файла может служить ключом к нему. Содержание текстового файла — его самое лучшее имя.

Отказ от использования иерархической файловой структуры не означает отказ от структурирования сохраняемой информации. Ничто не мешает вам создавать оглавления и указатели — но тогда структура будет частью вашего содержания, а не интерфейса. Такая структура не будет устанавливаться разработчиками, у которых могут быть другие идеи, отличные от ваших.

**Использовать каждый символ**

Программа должна взаимодействовать с пользователем на основе наименьшей единицы ввода. Пишешь с клавиатуры — познаково, говоришь голосом — пословно и т.д.

**Сделать универсальные команды**

Команды, которые должны быть доступны для пользователя в любой момент:

* проверить орфографию в текущей выборке;
* использовать текущую выборку как арифметическое выражение и вычислить его;
* передать текущую выборку по электронной почте;
* передать текущую выборку по факсу;
* перейти по данному URL;
* выполнить текущую выборку как программу, написанную на языке Java или любом другом.

**Не увлекаться пиктограммами**

Пиктограммы делают интерфейс более привлекательным в визуальном отношении и, при определенных условиях, могут способствовать большей понятности. Но иногда, вместо того, чтобы объяснять, пиктограммы зачастую сами требуют объяснений. К тому же, смысл пиктограмм часто зависит от культуры.

Количество пиктограмм, которые видно одновременно, не должно превышать 12.

Пиктограммы должны:

* визуально отличаться друг от друга
* хорошо отражать соответствующее понятие
* иметь разумно большой размер (обычно больший, чем могла бы быть текстовая надпись).

Пиктограммы нарушают принцип видимости, потому что смысл пиктограмм остается невидимым. Используйте пиктограммы только тогда, когда исследования показывают, что они эффективнее. Не бойтесь использовать текст.

**Исключить ошибки, а не предупреждать о них**

Когда вы пытаетесь написать текст сообщения об ошибке, пожалуйста, остановитесь и переделайте интерфейс таким образом, чтобы условие, при котором это сообщение об ошибке вызывается, не возникало.

Если отказаться нельзя, то сделайте сообщение об ошибке на полупрозрачном фоне. Так, пользователь увидит его и продолжит работу на компьютере, глядя на экран сквозь выданное сообщение. Сообщение закроется само, когда пользователь выполнит действие (перемещение курсора действием не считается).

*Если выдается сообщение, которое не требует ответа от пользователя, оно должно быть устранено.*

**Пусть потеет машина**

Не предоставляйте пользователю тех средств управления, которые должны работать всегда или никогда.

Если пользователь в следующий момент времени может выполнить только одно действие, пусть его выполнит компьютер.

**Критика — благо**

Если покупатель тратит свое время на тщательный анализ вашего продукта и делает конструктивные предложения для его улучшения, относитесь к этому с вниманием. Это нельзя рассматривать как попытку сделать вам выговор или нанести оскорбление. Этим он демонстрирует лояльность и интерес к вашему продукту.

**Упрощать аппаратное обеспечение**

Кабели должны присоединяться и отсоединяться от компьютера независимо от того, включен он или нет. Если их нельзя заменить без выключения питания, они становятся модальными.

**Кратко не значит понятно**

В большинстве современных интерфейсов акцент делается на краткость в ущерб ясности. Почему мы должны заниматься расшифровкой непонятного названия “Список” в выпадающем меню в текстовой программе, когда можно было бы использовать более понятное “Создать указатель или оглавление”.

Не следует путать простой внешний вид экрана с простотой использования интерфейса.

**Принципы отображения информации по Э. Тафти**

Визуальный дизайнер Эдвард Тафт (Edward Tufte, 1983, с. 105) разработал принципы отображения информации, среди которых первыми тремя являются следующие:

* данные следует показывать прежде всего остального;
* следует максимально увеличивать долю чернил, используемых для отображения данных;
* следует максимально уменьшать долю чернил, которые не используются для отображения данных.

Для того чтобы приложить эти принципы к устройствам с дисплеями, требуется всего лишь заменить в них слово чернила на слово пикселы. Серьезный, профессиональный пользователь желает, чтобы экраны были до отказа заполнены полезным содержанием. Экраны должны быть хорошо обозначены, снабжены простыми механизмами для осуществления поиска и получения информации, отражающей суть данного экрана. (В конце концов, раз уж мы сели за компьютер, мы должны извлечь из этого максимальную пользу.)

**Законы интерфейса**

**Первый закон интерфейса**

Компьютер не может причинить вред данным пользователя или своим бездействием допустить, чтобы данным был причинен вред.

**Второй закон интерфейса**

Компьютер не должен тратить впустую ваше время или вынуждать вас выполнять действия сверх необходимых.

**Зачем создавать качественный интерфейс?**

Тщательное проектирование и детальное определение технических и других требований не замедляют, а, наоборот, ускоряют процесс разработки. Создание качественного интерфейса полезно и с точки зрения долгосрочной перспективы, поскольку в результате приводит к:

* большей продуктивности работы пользователя;
* большему удобству для пользователя;
* большей ценности в глазах покупателя;
* уменьшению расходов на поддержку покупателей;
* ускорению и упрощению процесса внедрения;
* преимуществу перед конкурентами на рынке;
* лояльности к данной марке;
* упрощению инструкций и онлайновой помощи;
* более безопасным продуктам.

**Идеальный интерфейс по мнению Раскина**

Интерфейс с рабочим столом имеет довольно низкую эффективность, так как, находясь на его уровне, вы не можете выполнять свои задачи. В человекоориентированном интерфейсе, в котором нет ни рабочего стола, ни приложений, пользователь может в любой момент работать с необходимым ему содержимым.

Мы можем отказаться от использования файлов и файловых имен, оставив только одно пространство — пространство содержимого. В такой среде не нужно открывать или закрывать документы — вы просто уменьшаете масштаб системы до их размера и начинаете работать. Вам не нужно открывать приложения — вы просто копируете чистый документ. Игру не нужно запускать, вы просто масштабируетесь до нее.

Интерфейс должен быть по возможности самообучающимся. Это не означает, что интерфейс должен быть интуитивным, а означает, что пользователь может легко найти понятные объяснения или инструкции в тех случаях, когда он в них нуждается. Для упрощения процесса изучения при первой активизации продукта на экране должен отобразиться экран с текстом инструктивного содержания. Учебник и полное руководство по использованию должны быть частью интерфейса, доступной в любой момент.