**Дизайн в мире множества устройств. Введение**

В книге или журнале размер страницы не меняется. Мы не можем сделать книгу шире или уже — как она была свёрстана, так мы её и читаем. Но в интернете нет фиксированного размера холста. Сайт могут просматривать как на огромном мониторе, так и на мобильном телефоне. Поэтому сайт должен адаптироваться под размеры браузера: растягиваться, сжиматься, менять порядок и внешний вид элементов.

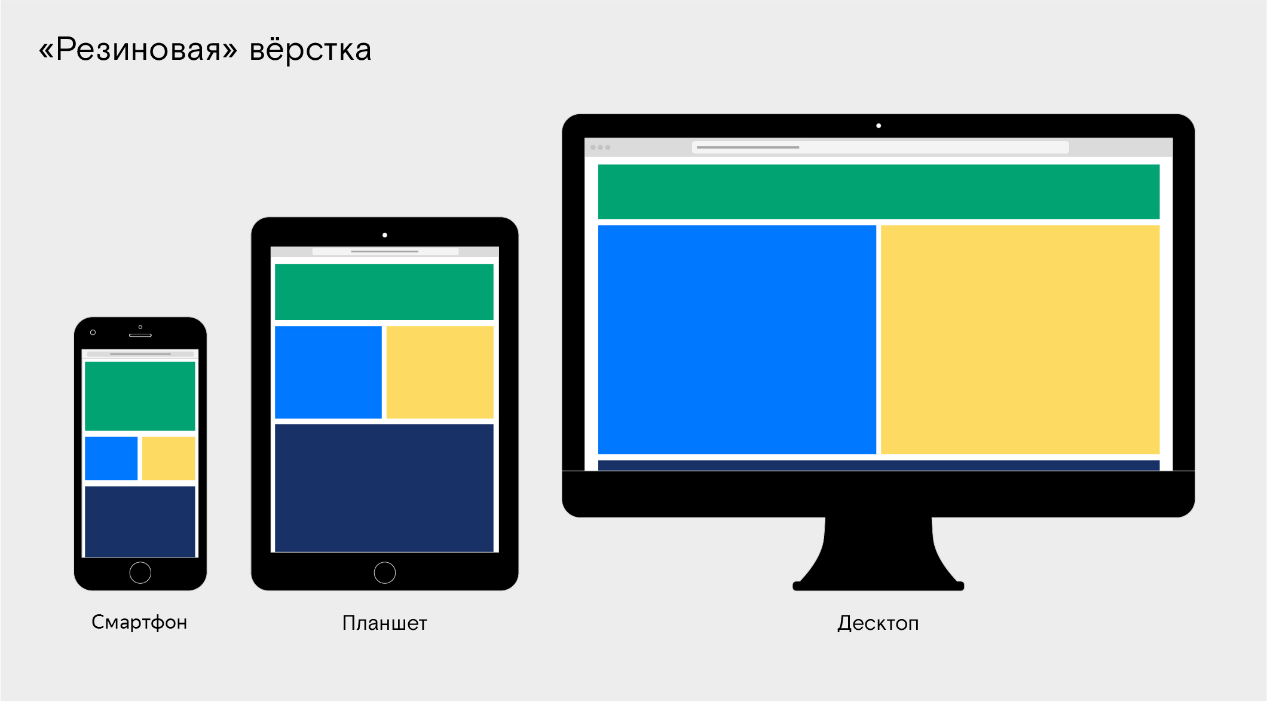
Когда смартфоны обзавелись первыми браузерами, разработчики отреагировали на это не сразу. Первым шагом было появление мобильных версий сайтов. Вы и сейчас можете встретить адреса вида mobile.company-name.com или m.company-name.com. Это совершенно другие сайты, которые загружаются, если пользователи приходят не с настольного компьютера или ноутбука. Такой подход безнадёжно устарел — CSS3 и HTML5 дали разработчикам широкий набор инструментов, чтобы преобразовывать страницы в разных условиях.

Дальше поговорим о подходах к построению интерфейсов, которые верстальщику важно знать. Это поможет вам правильно понимать задачу и выбирать инструменты для реализации макета. Начнём с теории о подходах к проектированию и дизайну, потом перейдём к практике построения адаптивных сеток и блоков.

# Разница между резиновой и адаптивной вёрсткой

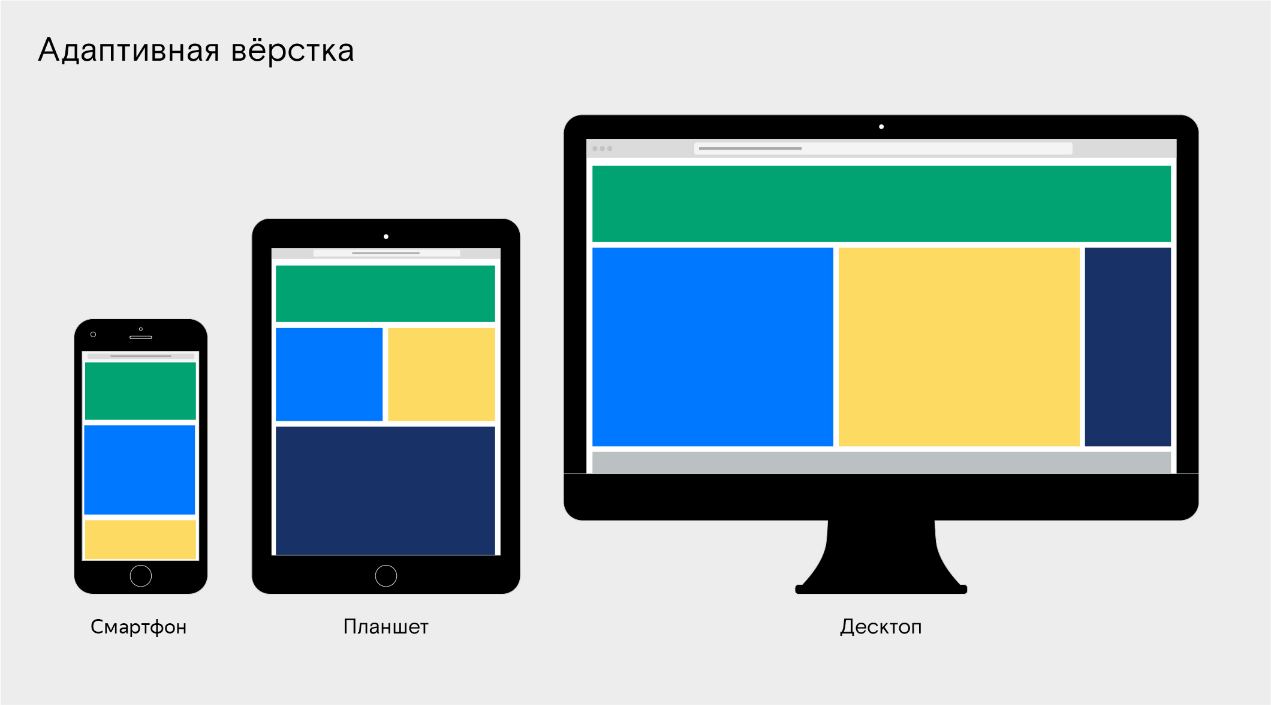
Есть два подхода к созданию интерфейса, подстраивающегося под разную ширину устройств: «резиновая» вёрстка и адаптивная вёрстка. Основная разница между ними в том, как интерфейс реагирует на изменение размеров окна браузера.

**При «резиновой» вёрстке** элементы растягиваются: пропорционально масштабируются под размеры окна, но не меняют своё положение и внешний вид.



Резиновая вёрстка. Элементы меняются в размере, но сохраняют своё расположение

**Адаптивная вёрстка** предполагает изменение внешнего вида элемента в зависимости от размера браузера.



Адаптивная вёрстка. Элементы меняют свою форму, размер и расположение

Секрет «резиновых» макетов в том, что размеры элементов задаются в относительных единицах измерения. В адаптивных макетах внешний вид элементов меняется по специальным правилам, прописанным для разных состояний браузера.

Существует термин «отзывчивый дизайн», который часто противопоставляют «адаптивному дизайну». При любом из подходов сайт верстают в нескольких состояниях: для телефона, планшета, ноутбука, десктопа. Но если менять размер окна браузера, отзывчивый сайт переходит из одного состояния в другое плавно, а при адаптивном — скачками.

Без применения JavaScript есть только три способа отреагировать на изменение размера окна браузера:

* применяя относительные единицы измерения, сделать элемент резиновым, задав его размеры в относительных единицах: %, vh, vw;
* писать CSS-правила, срабатывающие при разных условиях: одна ширина элемента при одном размере окна, другая — при другом;
* показать или скрыть элемент при определённых условиях. Это частный случай применения второго способа, например значение свойства display меняют на none и наоборот).

Сначала мы разберём, как сделать сетку резиновой, а в конце спринта расскажем, как задавать стили, срабатывающие в зависимости от обстоятельств.

# Мобильный или десктоп: с чего начать вёрстку?

Мы разобрались, что в зависимости от размера экрана устройства, интерфейс может выглядеть по-разному. Закономерный вопрос: какой вид интерфейса брать за исходную точку, с чего начинать вёрстку? Единого мнения нет — но есть два подхода.

## Desktop First

Традиционный подход — начинать с десктопа. Сначала верстается интерфейс под настольные компьютеры и ноутбуки, потом — под планшеты и смартфоны. Отдельно описывают стили для очень больших мониторов.

Так сложилось исторически, ведь дизайн многих сайтов был уже готов, когда появились первые смартфоны с полноценными браузерами.

Этот подход используют чаще, чем альтернативный. Многим дизайнерам так работать привычнее. И согласовывать макеты с заказчиками проще.

## Mobile First

На многие сайты чаще заходят с телефона, чем с компьютера: социальные сети, интернет-магазины, сервисы заказа еды. Дизайн таких сайтов продумывают так, чтобы он был удобен в первую очередь на телефоне. Сначала пишут стили для самого маленького экрана, потом дорабатывают их для устройств побольше.

Минимальный продукт при таком подходе производится быстрее, ведь на смартфоне элементов обычно меньше, анимации сдержаннее или отсутствуют вовсе.

Оба подхода имеют право на жизнь. Споры о том, как верстать правильно, не угасают. Наша задача — разобраться с достаточным количеством инструментов, чтобы уметь делать сайты любым из подходов.

**Адаптивный макет**

В начале проекта по вёрстке вы, скорее всего, получите макет дизайна для разных устройств. Например, как в [проектной работе этого спринта](https://www.figma.com/file/OyRWEjU6wBwRe1hapzQoLx/Sprint-3%3A-Russia-%2F-desktop-%2B-mobile). Для демонстрации поведения интерфейса на разных устройствах в основном выбирают самые популярные разрешения — 1440, 1280, 1024, 768 пикселей и минимальный — 320 пикселей.

Так много примеров будет не всегда, дизайнеры часто ограничиваются тремя: одно для настольного компьютера или ноутбука и по одному на планшет и смартфон. Поведение в промежуточных разрешениях вам нужно придумать самому или обсудить с дизайнером. При вёрстке важно, чтобы сайт не ломался на промежуточных разрешениях, ведь вы не знаете, ширину устройства пользователя. Посмотрите на [статистику использования различных разрешений мониторов](https://www.screensizemap.com/). Если вам стало не по себе от такого разнообразия, это нормальное чувство.

Чтобы добиться хорошего результата вёрстки, нужно следовать трём правилам:

* создавать «резиновую сетку»;
* создавать «резиновые блоки»;
* изменять CSS-стили в контрольных точках.

Вам уже знаком один из способов строить резиновые сетки — это флексбокс. Он позволяет настраивать поведение элементов при изменении ширины устройства. Но в этом спринте вы пополните свой арсенал куда более гибким инструментом — grid layout. С него мы начнём практическую часть занятий.

В последней теме спринта мы обобщим всё, что рассказывали о создании резиновых блоков.

Изменять CSS-правила в зависимости от ширины устройства тоже научимся в последней теме, разобрав директиву @media. Именно она отвечает за реализацию адаптивности, а не «резины».

Из такого многообразия инструментов важно выбрать наиболее подходящий. Анализируя макет, обращайте внимание:

1. Как меняется количество строк и колонок при сжатии элементов?
2. Меняют ли элементы свой порядок относительно друг друга?
3. Как меняются размеры и отступы? Попробуйте найти или предположить закономерности.
4. Какие элементы пропадают и появляются на разных разрешениях?
5. Как изменение элементов связано с изменением стилей их контейнеров? Например, свойство flex-direction часто меняют с row на column, а спозиционированные элементы возвращаются в поток.

Ответы на эти пять вопросов — уже половина дела. Дальше обсудим варианты реализации.

# Неполный бриф, неточный макет

Почти каждый слышал истории о шаровых молниях: как та влетела в форточку, проплыла мимо замерших родственников и друзей семьи и вылетела обратно в окно. Интересно, что приборы метеорологов ни одной шаровой молнии не фиксировали. Примерно так же обстоят дела с «идеальным брифом».

Полных брифов и макетов не бывает никогда. Всегда можно что-то улучшить: подготовить картинки в точности под размер, описать для каждого элемента и окна, какие для него задействованы анимации и ховеры, показать изменение вёрстки перед, во время и после брейкпоинта. Всё это займёт массу времени дизайнера, чтобы отобразить в макете, но можно уточнить за полчаса личного общения.

Без некоторых данных работа встаёт, сроки горят, проект не заканчивают в срок. Как с этим бороться? Постараться до старта работы задать все ключевые вопросы. Ниже мы подготовили список вопросов, которые часто нужно выяснить заранее.

### Что нужно знать заранее

* В каких браузерах проект должен работать? От этого зависят технологии, которыми вы сможете пользоваться.
* На каких устройствах и экранах проект должен работать? Чаще всего это мобильные телефоны, планшеты и десктоп.
* Как должна вести себя вёрстка в пограничных значениях: очень большой десктоп, очень маленький мобильный? Что происходит с вёрсткой в промежутках между «брейкпоинтами» — точками, где вёрстка «переламывается», и вступают в силу правила для другого разрешения экрана.
* Нужна ли заказчику админка — панель управления контентом сайта, доступная для владельца и модераторов? Если нужна, скорее всего, понадобится отдельный бэкенд-разработчик, который будет её делать. Это значительно увеличивает стоимость проекта. Если заказчику админка не нужна, значит разработка будет дешевле и проще.
* Если вы берёте проект на фриланс, обязательно договоритесь с заказчиком, сколько этапов правок вы будете делать. Без ограничений правки не закончатся никогда. Либо определите дополнительные условия работы над правками. Например, с почасовой или сдельной ставкой.

### Чего часто может не хватать в макете или брифе

* Объяснения, как и где работают анимации и ховеры — эффекты при наведении на ссылку или картинку. Это касается кнопок, выезжающих меню, попапов и ссылок.
* Файлов со шрифтами. Также полезно проверить лицензию на их использование. Для этого просто вбейте в гугле слово typeface и название шрифта, и посмотрите по первым результатам в выдаче на раздел «License Information». Если лицензия не открытая (например, Commercial Licenses), то стоит задать этот вопрос дизайнеру.
* Текстов и картинок. Убедитесь, что у вас есть тексты и картинки, которые участвуют в вёрстке.

Это минимальный набор того, что следует проверить после изучения макета и брифа. С опытом вы научитесь это делать самостоятельно и быстро.