1. Что такое прототипирование?

Прототипирование — это процесс создание интерактивного опыта, ко-торый включает создание прототипа на основе разработанных мокапов или вайрфреймов.

1. Что такое вайрфрейм?

Вайрфрейм (каркас, блочная схема) — это образ продукта низкой точности, с упором на структуру и содержание, схематичное изображение, набросок пользовательского интерфейса будущего продукта. Это первый шаг в процессе прототипирования.

1. Какие есть виды вайрфреймов?

Виды вайрфреймов:

1. Вайрфреймы низкой точности (Low-fidelity). Напоминает грубый набросок или быстрый макет, каркасы низкой точности имеют меньше деталей и быстро создаются. Они более абстрактны, используют прямоугольники и маркировку для представления содержимого приложения. Содержат фиктивный контент, контуры элементов, какие-то образцы текста или символический контент. Например, вместо реальных изображений можно использовать прямоугольник-заполнитель. Пример вайрфрейма низкой точности представлен на рисунке 1.
2. Вайрфреймы высокой точности (High-fidelity). Они включают уровень детализации, который более точно соответствует дизайну фактической деятельности приложения. Вайрфреймы высокого качества включают в себя более реальный контент, оперативный выбор типографики и информацию о размерах изображения. Пример вайрфрейма высокой точности представлен на рисунке 2.

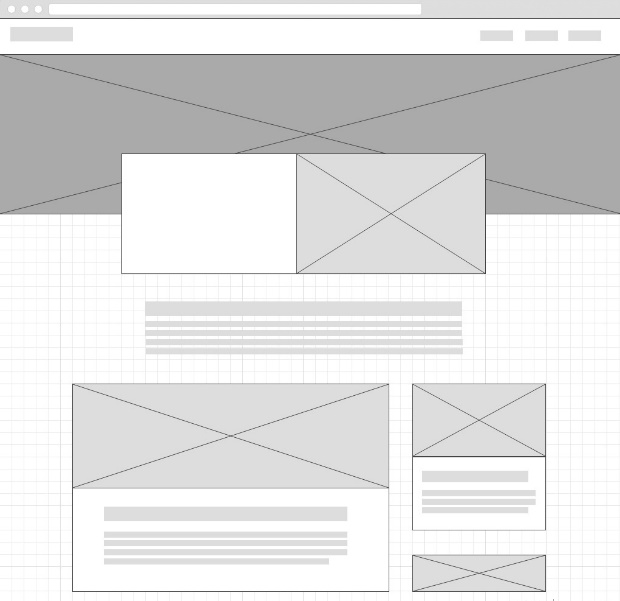


Рисунок 1 – Пример вайрфрейма низкой точности

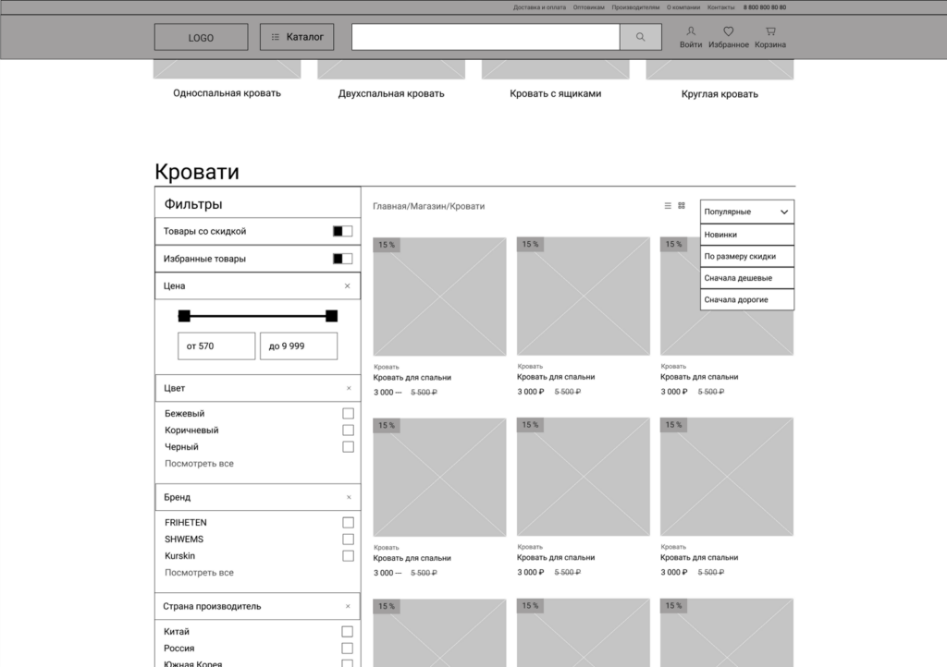


Рисунок 2 – Пример вайрфрейма высокой точности

1. Для чего необходимы вайрфреймы?

Вайрфреймы играют ключевую роль в процессе разработки, так как они позволяют команде разработчиков и дизайнеров легко обмениваться идеями и принимать решения о том, как лучше всего реализовать функциональность и взаимодействие элементов интерфейса. Они также помогают уточнить требования заказчика и снизить риски возможных недоразумений.

С помощью вайрфреймов можно быстро прототипировать интерфейсы и проводить тестирование концепции продукта на ранних стадиях разработки, что позволяет обнаружить и исправить проблемы до того, как они станут критическими и могут повлиять на пользовательский опыт.

1. Что такое прототип?

Прототип — это средне или высоко детализированное представление конечного продукта, которое имитирует взаимодействие пользователя с ин-терфейсом. Прототипы всегда интерактивные.

1. Для чего необходим прототип?

Прототип необходим для проверки логики функционала. Это симуля-ция взаимодействия между пользователем и интерфейсом. В прототипах при-сутствует функционал, и пользователь может выполнять некоторые действия: переходить между экранами/страницами продукта, нажимать на кнопки, от-крывать модальные окна (обычно это просто создание кликабельности в ре-дакторе для прототипирования).

1. Чем прототип отличается от вайфрейма?

Может добавлять интерактивности к вайрфреймам или к макетам. В от-личие от того, будет определяться его степень детализации. Обычно прототип добавляется уже поверх вайрфрейма и будет являться серединой на пути к высококачественному изображению финального продукта.

Таким образом под прототипом понимается динамическая интерактив-ная модель продукта.

1. Что такое мокап (макет)?

Мокап (mockup или mock-up — макет) является полноцветным и дета-лизированным наброском дизайна. Это живописный аналог вайрфрейма. В нем уже присутствует полная палитра оттенков, конкретные изображения элементов (кнопок, иконок, шапки сайта), фон страниц, текстовые шрифты и прочие графические элементы. Часто мокапы это актуальный графический дизайн.

1. Перечислите правила чистого UI-дизайна.
2. Простые шрифты. Использование простых и минималистичных шрифтов сделает тексты и весь дизайн более читабельными. Пример представлен на рисунке 7 (шрифты в первой колонке лучше использовать минимально раз для каких-то крупных заголовков).

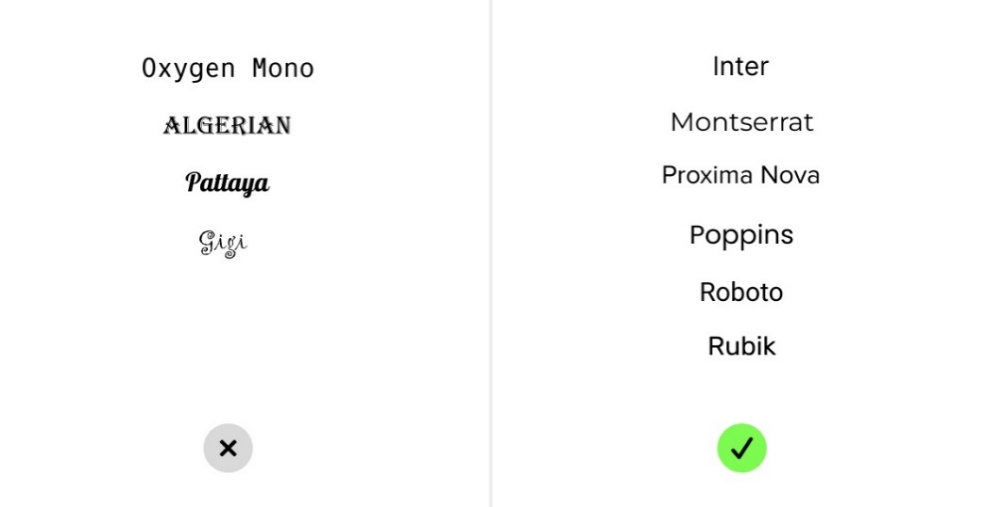


Рисунок 7 – Пример правила использования простых шрифтов

1. Необходимо выравнивать элементы. Правильные отступы и интервалы создают чистый и упорядоченный макет, что упрощает и ускоряет восприятие информации. Пример представлен на рисунке 8.

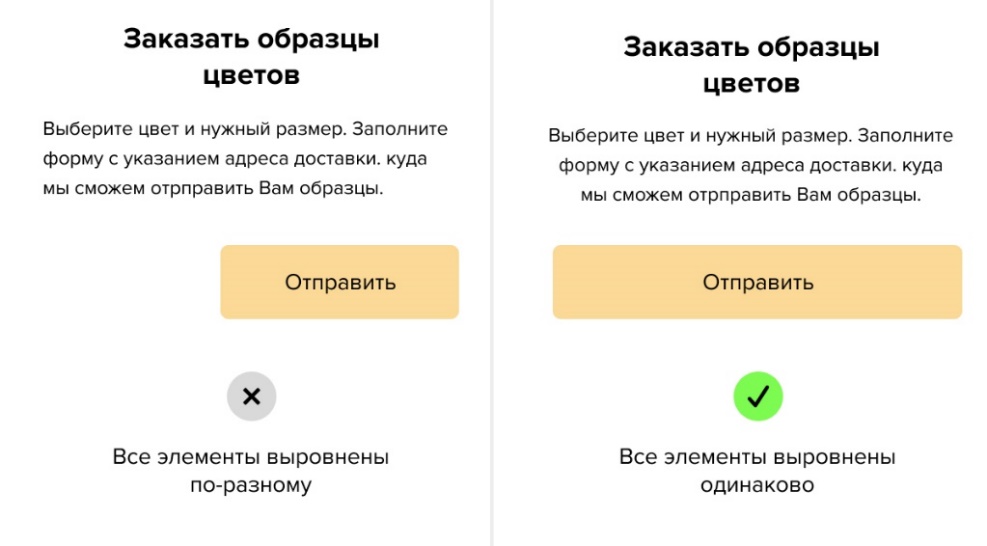


Рисунок 8 – Пример выравнивания элементов

1. Не использовать грязные тени. Рекомендуется использовать более светлые оттенки чёрного, чтобы убрать с тени напряжение и сильный контраст. Пример показан на рисунке 9.

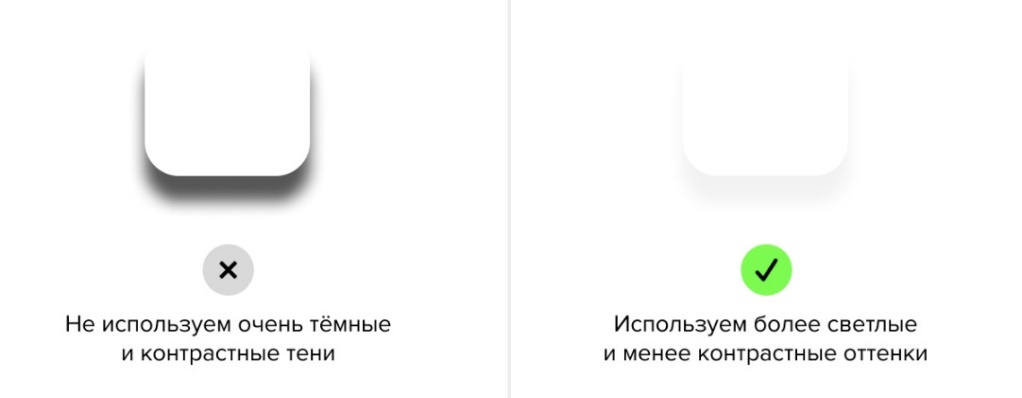


Рисунок 9 – Пример правила использования теней

1. Больше свободного пространства. Необходимо предоставлять элементам больше свободного пространства для того, чтобы они смогли дышать. Это способствует облегчению навигации. Пример представлен на рисунке 10.

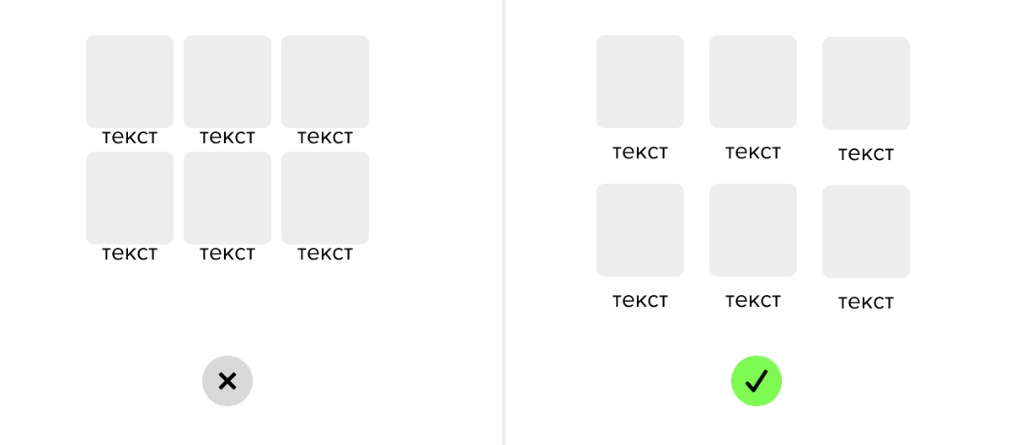


Рисунок 10 – Пример правила использования свободного пространства

1. Использование нужных и полезных элементов. Ограничение количества элементов в одном пространственном блоке повышает концентрацию пользователя на необходимом. Также необходимо использовать правила композиции при группировке элементов. Пример представлен на рисунке 11.

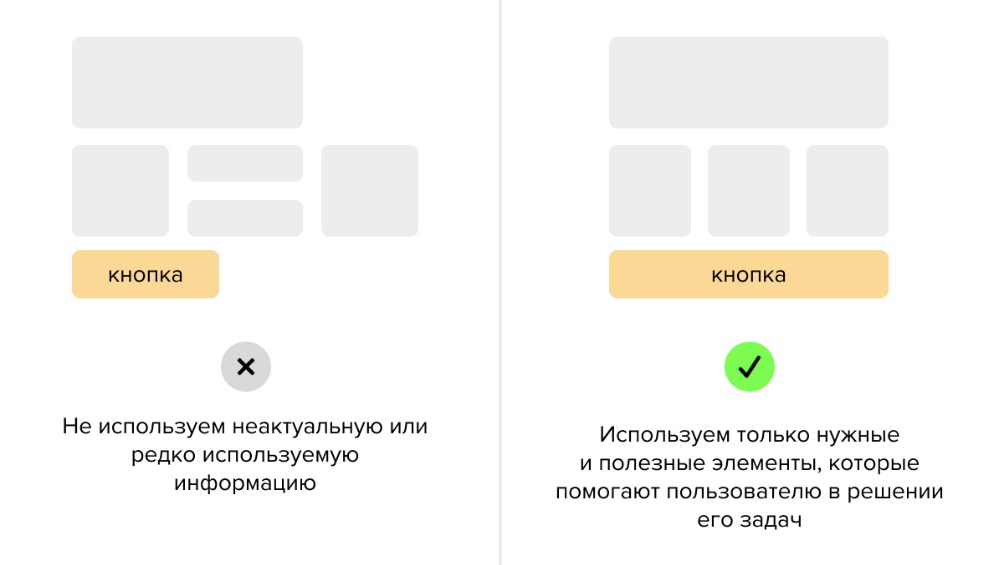


Рисунок 11 – Пример правильного использования элементов

1. Более простой дизайн. В большинстве случаев наличие большого количества эффектов и стилей портит пользовательский интерфейс. Рекомендуется не делать слишком много эффектов, например, так как тени, градиенты, жирные обводки элементов. Пример показан на рисунке 12.

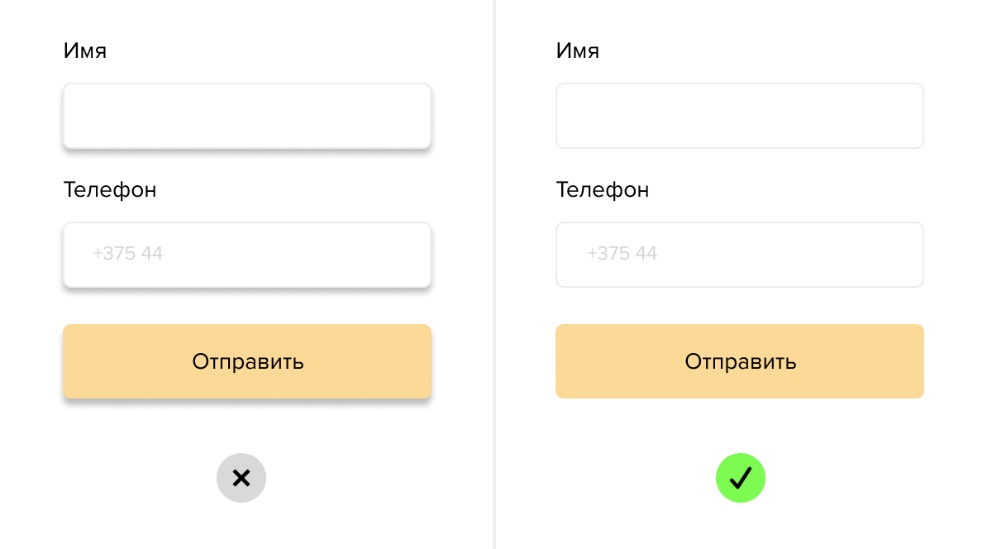


Рисунок 12 – Пример правильного использования эффектов

1. Баланс и выравнивание. Необходимо всегда помнить, что у всех элементов в интерфейсе есть вес. Вес можно определить по цвету, размеру, текстуре. Необходимо соблюдать баланс, правила композиции и правильно выравнивать элементы.
2. Быть последовательным и соблюдать единообразие в интерфейсе. Согласованный единообразный пользовательский интерфейс подразумевает использование одинаковых шаблонов проектирования, одинаковые элементы, отступы, а также согласованные команды во всём интерфейсе. Пример представлен на рисунке 13.

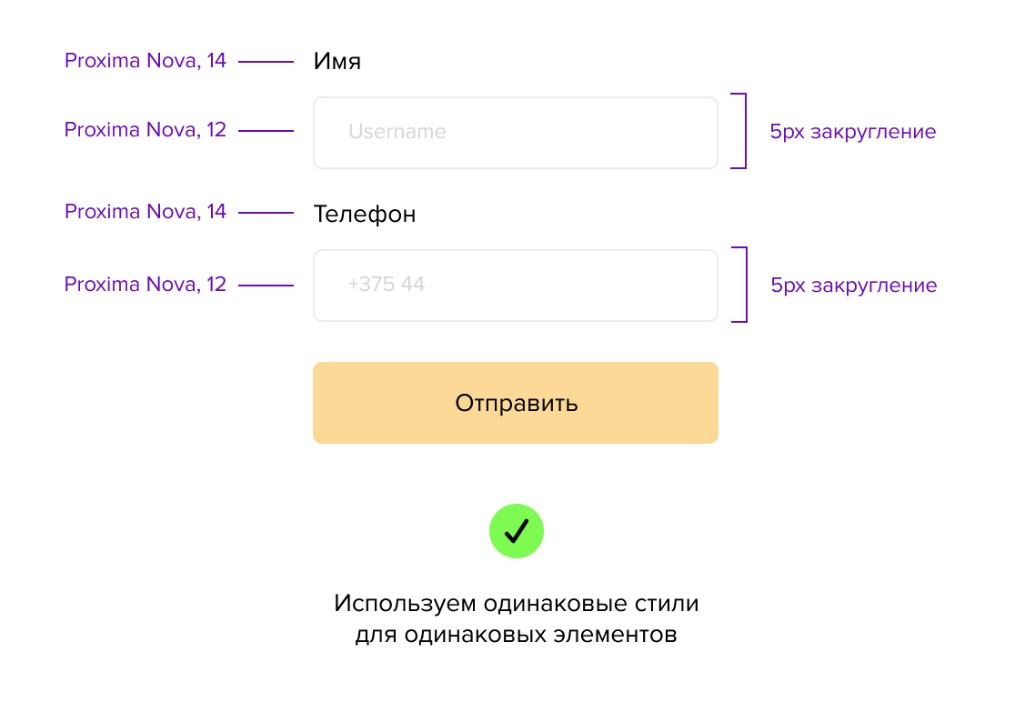


Рисунок 13 – Пример единообразного интерфейса

1. Единая система шрифтов и начертаний. Необходимо использовать одну систему шрифтов, начертаний, толщины для основного текста, заголовков, подзаголовков и так далее. Даже небольшая разница в размере между двумя типами шрифта может исказить контрасты и иерархию.
2. Читаемые заголовки. Необходимо делать заголовки более акцентными и контрастными, для того, чтобы направить взгляд пользователя от наиболее важных вещей к менее важным. Пример показан на рисунке 14.

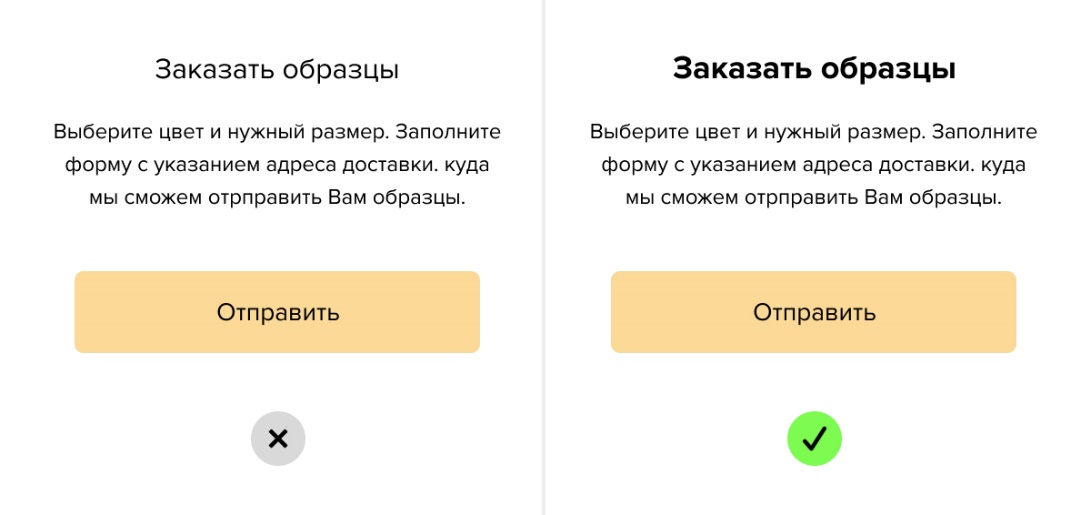


Рисунок 14 – Пример создания читаемых заголовков

1. Визуальная иерархия элементов. Строгая визуальная иерархия — основной принцип успешного пользовательского интерфейса. Состоит в расположении элементов интерфейса таким образом, чтобы объяснить уровень важности каждого элемента и направить пользователей к выполнению желаемого действия. Пример представлен на рисунке 15 (дизайн не путает пользователя и создаётся иерархия, в которой есть призыв к целевому действию).

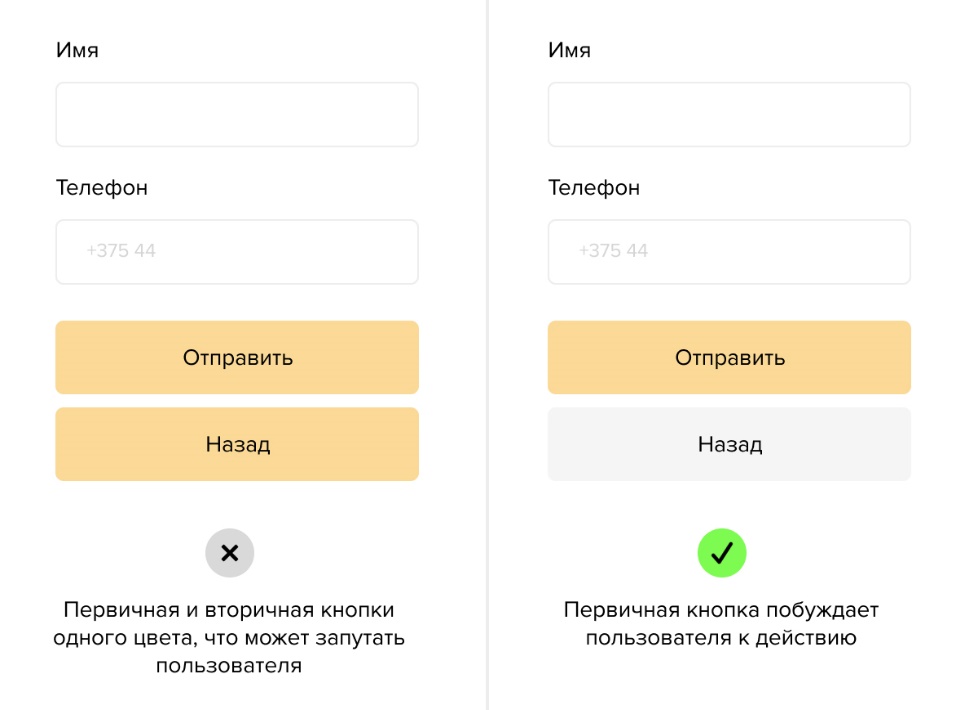


Рисунок 15 – Пример создания правильной иерархии

1. Делать различными placeholders, labels и введённые пользователем данные. Пример представлен на рисунке 16.

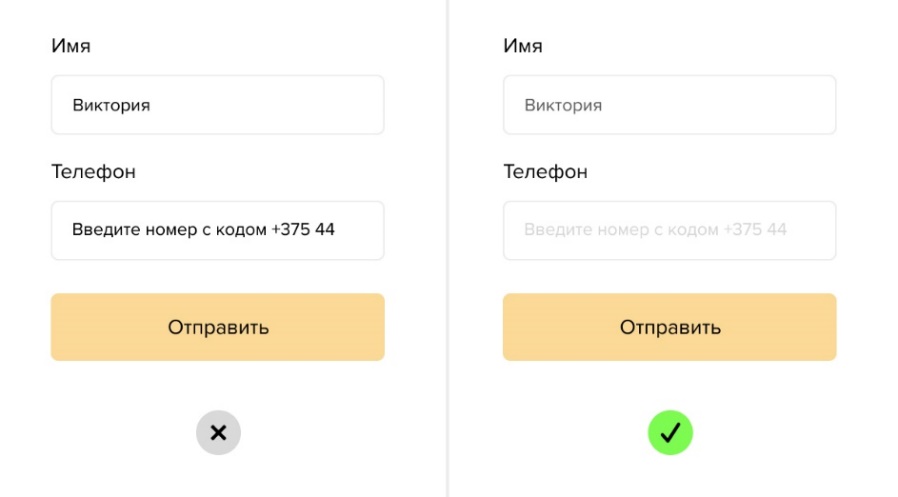


Рисунок 16 – Пример создания элементов формы

1. Использовать одну систему отступов/интервалов. Рекомендуется использовать одну систему отступов. Вокруг логических блоков должны быть установлены отступы одинакового размера (например, вверху и внизу, а также слева и справа). Пример представлен на рисунке 17.

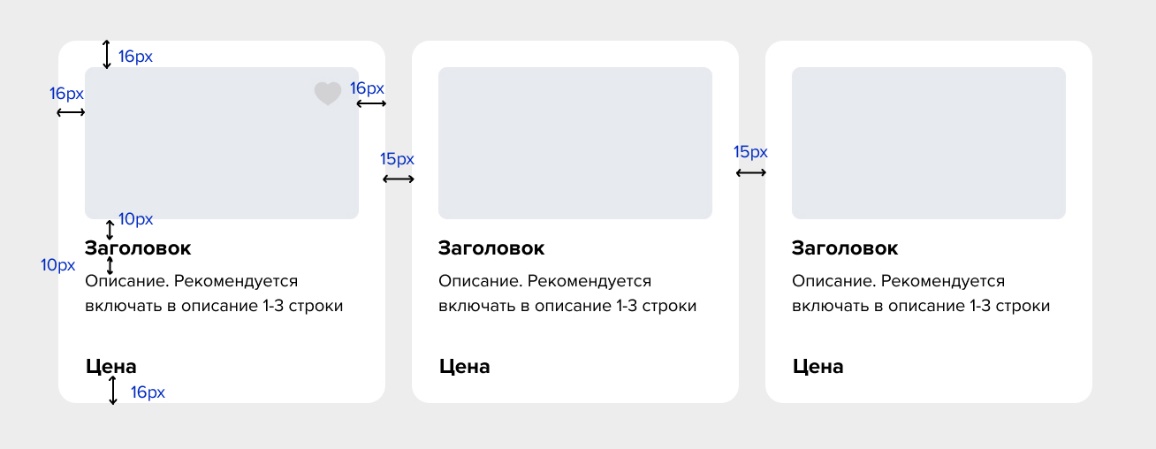


Рисунок 17 – Пример использования единой системы интервалов

1. Связанные элементы должны быть расположены ближе друг к другу. И наоборот, несвязанные элементы должны быть расположены подальше друг от друга. Пример представлен на рисунке 18.



Рисунок 18 – Пример связанных элементов

1. Элементы одного «типа» должны быть расположены на одинаковом расстоянии друг от друга. Пример представлен на рисунке 19.



Рисунок 19 – Пример расположения элементов одного «типа»

1. Более крупные элементы должны иметь больше свободного пространства вокруг себя.
2. Пространство между различными логическими блоками должно быть больше, чем пространство между заголовком и текстом внутри каждого блока.
3. Рекомендуется использовать 3 размера шрифта: для заголовка, подзаголовка и основного текста. При помощи разного размера, толщины и стиля шрифтов, можно направить внимание пользователя к важным элементам.
4. Более важные элементы рекомендуется увеличивать, а менее важные – уменьшать. Главное, чтобы крупных элементов на один компонент было не больше двух.
5. Не забывать про паттерны сканирования. Необходимо помнить, что существуют две главные траектории, по которым человек считывает информацию — F-паттерн и Z-паттерн. Ключевой контент должен располагаться именно по этим маршрутам. Пример показан на рисунке 20.

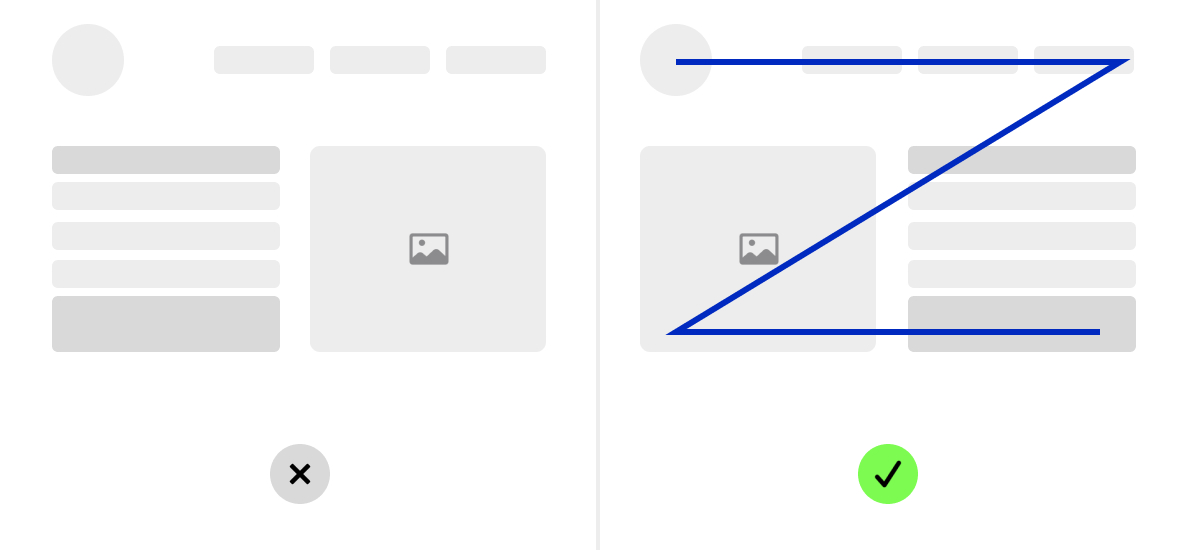


Рисунок 20 – Пример использования Z-паттерн

1. Не использовать placeholder в качестве label. Пример представлен на рисунке 21.

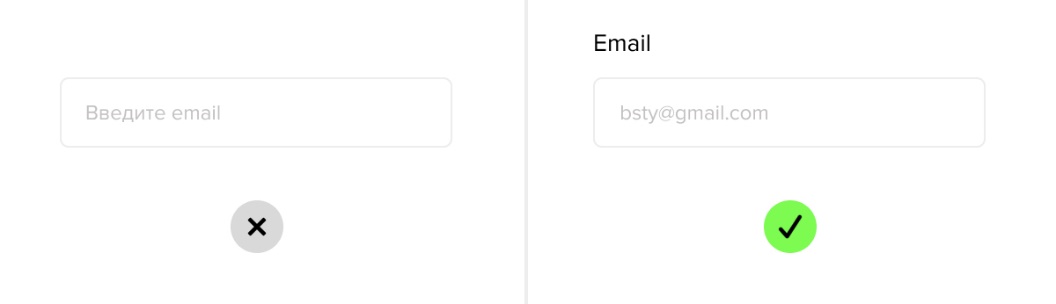


Рисунок 21 – Пример правильного использования placeholder и label

1. Добавлять свободное пространство кнопке. Пример показан на рисунке 22.



Рисунок 22 – Пример правильно спроектированной кнопки

1. Рекомендуется использовать формы, состоящие из одной колонки пример показан на рисунке 23, либо, в котором две колонки, но порядок размещения полей идёт по горизонтали, пример показан на рисунке 24.

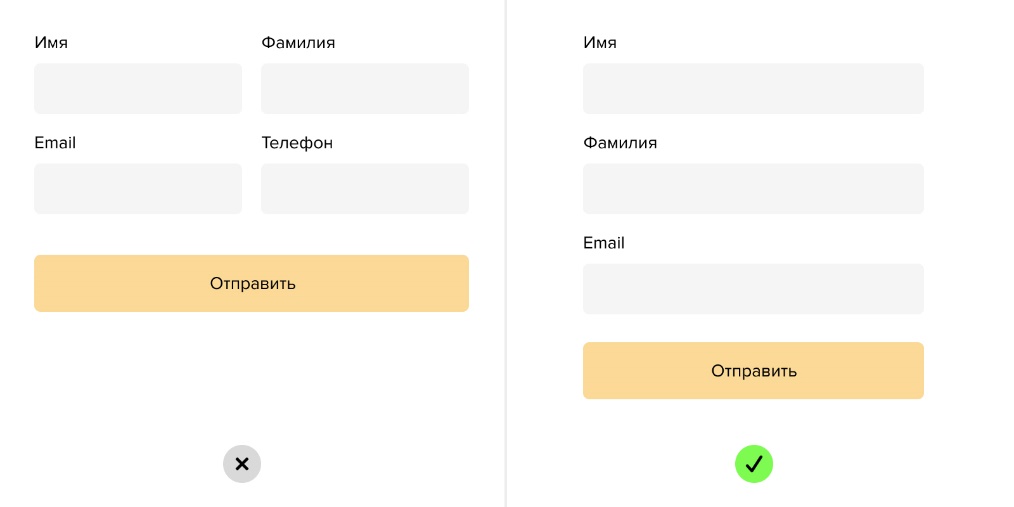


Рисунок 23 – Пример формы, состоящей из одной колонки

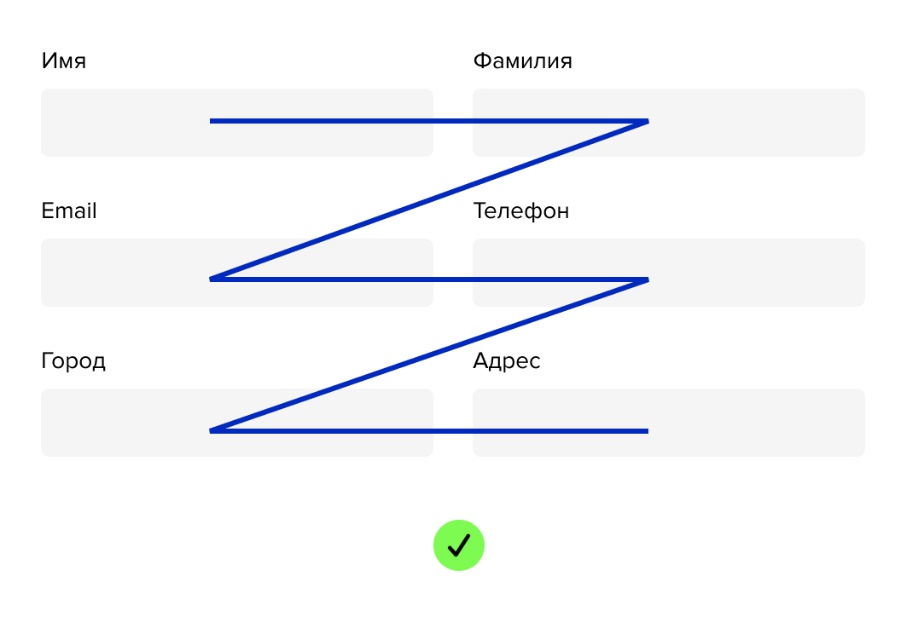


Рисунок 24 – Пример формы, где поля размещены по горизонтали

1. Рекомендуется не использовать символы только верхнего регистра в кнопках. Пример показан на рисунке 25.



Рисунок 25 – Пример правильно оформленного текста внутри кнопки

1. Основной контент страницы/экрана должен располагаться в пределах сетки и быть выровнен по ней. Пример показан на рисунке 26.

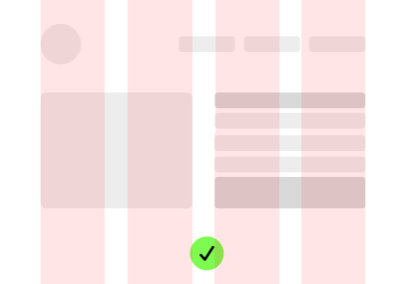


Рисунок 26 – Пример выравнивания контента внутри сетки