Дизайн UI/UX

О курсе

- Дизайн UI/UX
  - Основы и этапы
  - Принципы и применяемые технологии
  - Используемые инструменты
  - Основной ресурс
    - https://www.uprock.ru

О курсе

- JavaFX
  - Назначение и возможности
  - Установка инструментария
  - Архитектура JavaFX-интерфейса
  - Базовые классы
  - Основные ресурсы
- https://openjfx.io/
- https://gluonhq.com/products/javafx/
- https://metanit.com/java/javafx/1.1.php

UI и UX

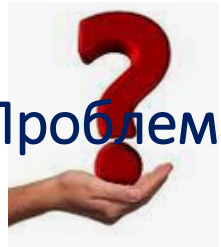
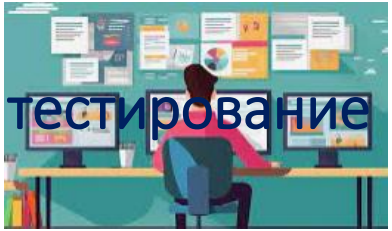
- UX (User Experience) – анализ предпочтений пользователей ресурса и превращение его в продукт
- UI (User Interface) – оформление этого продукта, делает его эстетически приятным

Человечность дизайна выражается в том, чтобы подстроить предмет дизайна под человека с его слабостями, а не заставлять человека подстраиваться под предмет.

Илья Бирман

UX - дизайн

- Разработка каркаса ресурса на основе анализа предпочтений пользователей

Проблема ->  Работа с ЦА ->  тестирование -> 

Каркас/прототип 

Задачи UX-дизайна

- Создание и согласование плана разработки
- Разработка информационной архитектуры
- Выяснение потребностей целевой аудитории и создание пользовательского потока
- Разработка каркаса интерфейса
- Тестирование

UI - дизайн

- Использование элементов визуального дизайна для удобного взаимодействия пользователя с разрабатываемым продуктом
- Сделать каркас эстетически привлекательным и оптимизированным для выполнения на разных устройствах

Задачи UI-дизайнера

- Разработка макета и всех визуальных компонентов интерфейса (цвета, шрифты, заначки, кнопки и пр.)
- Согласовывать удобство использования продукта с брендом
- Адаптация макета ко всем устройствам, платформам и размерам экрана по форме и по функциям
- Разработка переходов, включение анимации и интерактивных элементов

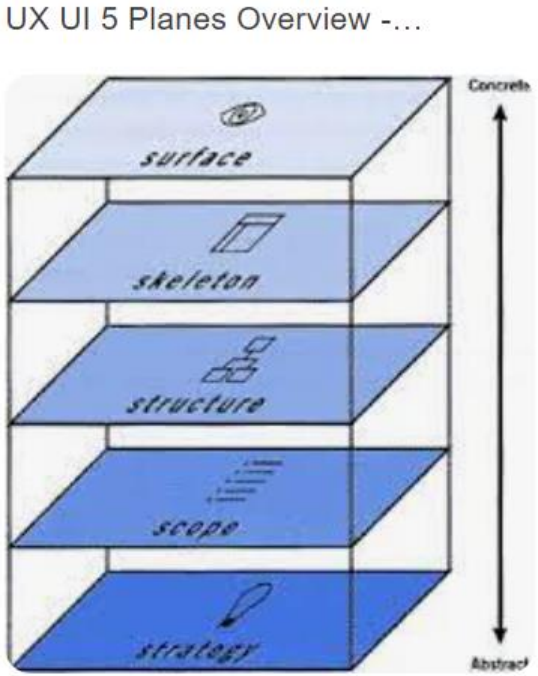
Порядок разработки по принципам UI/UX дизайна

- Сбор информации (опросы или интервью)
- Аналитика(изучение аналогов)
- Разработка структуры и карты пользовательских путей
- Формирование прототипа
- Работа с контентом (фото, видео, инфографика)
- Разработка визуальной оболочки (UI)
- Тестирование

Слои UX

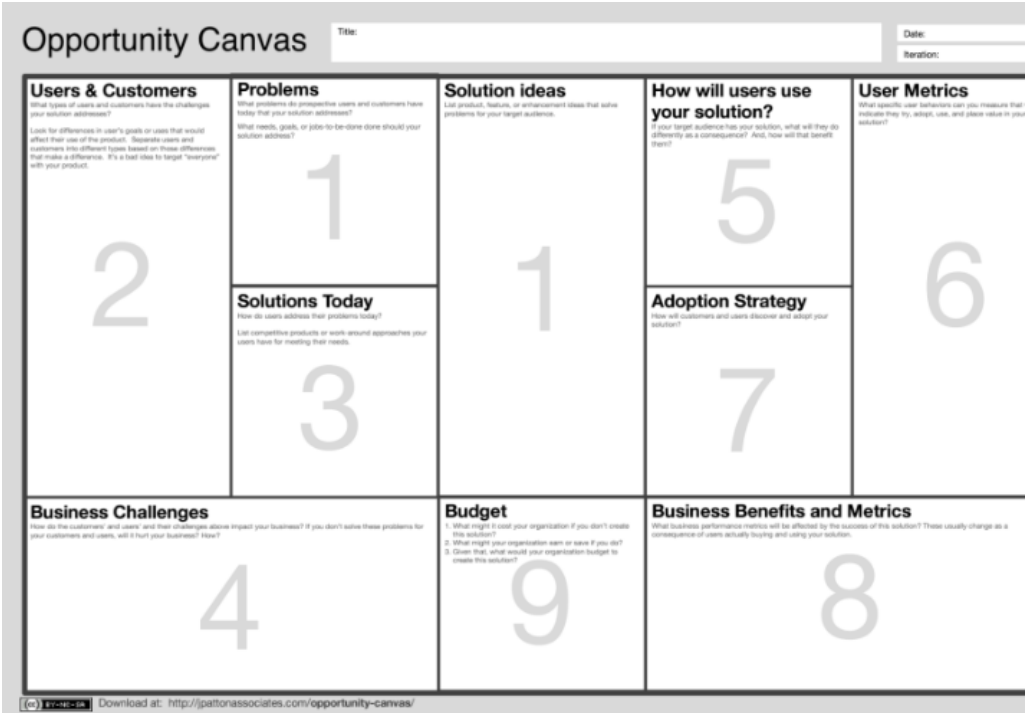
5 слоев дизайна (Джесс Гарретт 2001)

1. Поверхность
2. Компоновка
3. Структура (информационная архитектура)
4. Скоуп (функции)
5. Стратегия



Strategy Plane

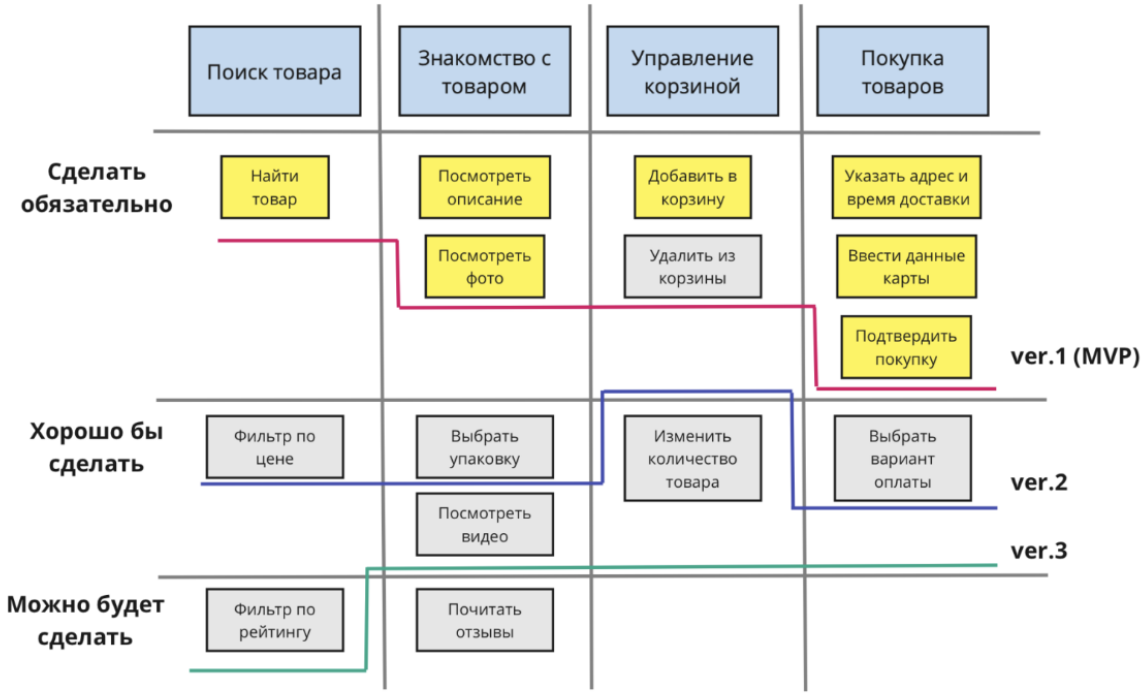
- Зачем создается?
- Кто будет его использовать?
- Чем отличается от аналогичных ?
- Какая возможна цена издержек?



Scope Plane

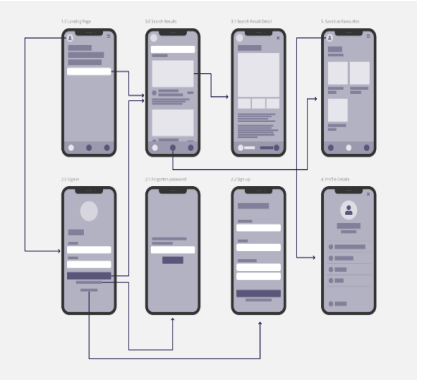
- Сортируем функции по приоритетности (L,M,H)
  - Важность для пользователя
  - Важность для проекта (бизнеса)
  - Трудоемкость в разработке
- Распределяем на этапы реализации
- Определяем функции, включаемые в релиз (Minimal Viable Product & Minimal Viable Release)
- Декомпозиция на основе пользовательских историй (User Story Mapping)

Scope Plane


Поиск товара
Знакомство с товаром
Управление корзиной
Покупка товаров
Сделать обязательно
Найти товар
Посмотреть описание
Добавить в корзину
Указать адрес и время доставки
Посмотреть фото
Удалить из корзины
Ввести данные карты
Подтвердить покупку
ver.1 (MVP)
Хорошо бы сделать
Фильтр по цене
Выбрать упаковку
Изменить количество товара
Выбрать вариант оплаты
ver.2
Посмотреть видео
ver.3
Можно будет сделать
Фильтр по рейтингу
Почитать отзывы

Structure Plane

- Расставляем функциям приоритеты. Минимизация шагов для самых востребованных
- Разработка навигации по экранам
- Внутренняя навигация
- Построение карты шагов при использовании функции
- Построение потока экранов

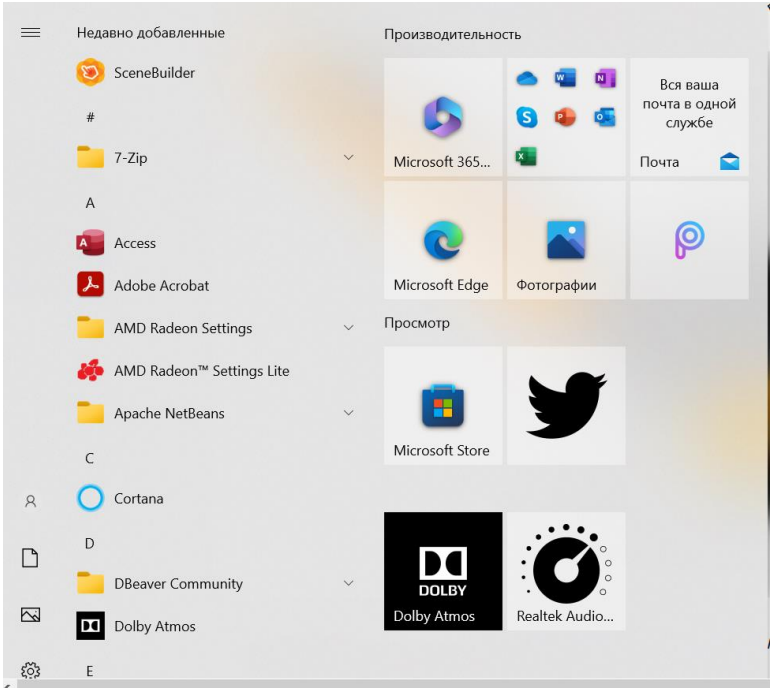


Skeleton Plane

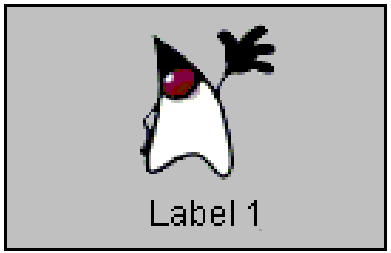
- Взаимное расположение элементов интерфейса
- Контрастность элементов
- Учитываемые факторы
  - Точка старта (eyecatcher)
  - Иерархия и контрастность (цвет, размер, движение, форма)
- Привычность (предсказуемость) расположения
- Паттерн считывания (Z/F)
- Когнитивная нагрузка (изображения, пиктограммы, минимизация текста, структурирование)

Surface Plane

- Определение стиля элементов и иллюстраций
- Компоновка

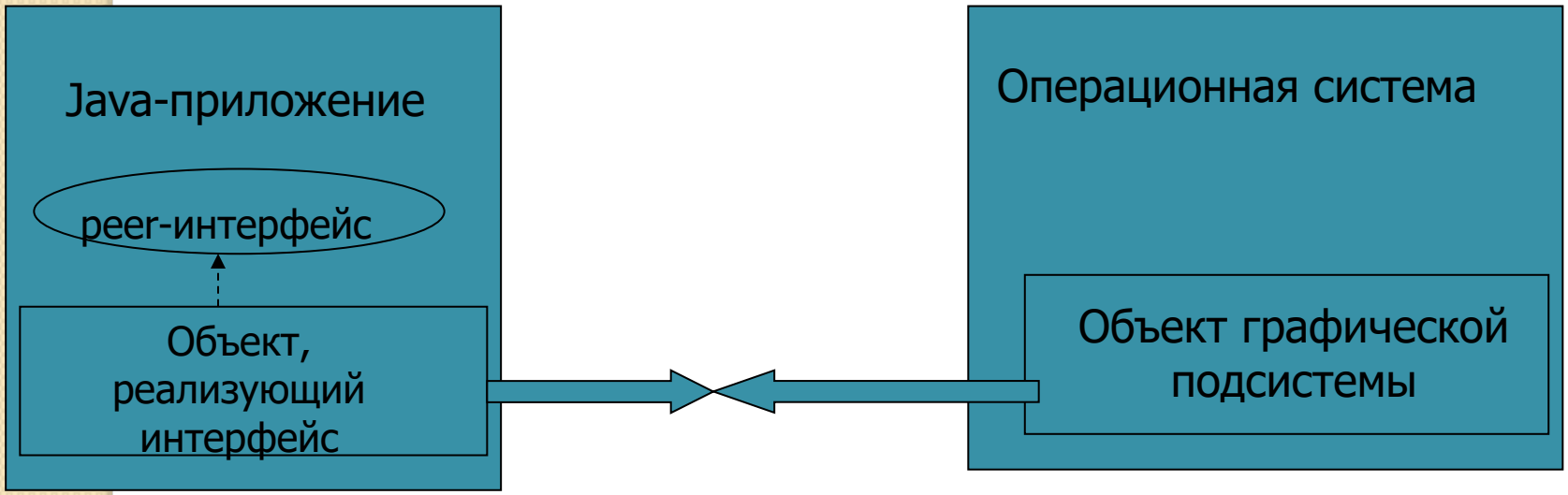


Средства создания графического интерфейса в Java



Библиотека AWT

- peer-to-peer интерфейсы («тяжелые» компоненты)



«Легкие» (lightweight) компоненты

- Сохранение заданного при создании вида (look and feel)
- Возможность изменить вид в любой момент работы приложения (PL&F)
- Библиотека «легких» компонентов Swing

JavaFX

- 2007г. – альтернатива Flash (Sun Microsystems)
- JavaFX 8 вошла в JDK(Oracle)
- Использует язык разметки FXML
- Начиная с Java 11 версии, The Java FX не входит в JavaSE (компания Gluon)

- Ресурсы

  https://openjfx.io/

  https://gluonhq.com/products/javafx/

  https://habr.com/ru/articles/474292/

Приложение JavaFX

```
public class JavaFX1  extends Application{
    public static void main(String[] args) {
                launch(args);
    }
    @Override
    public void start(Stage stage)  {
      Label lb = new Label("Hello!!!");
      Pane root= new Pane(lb);
      Scene scene = new Scene (root,300,250);
      stage.setScene(scene);
      stage.setTitle("javaFX application");
      stage.show();
    }
}
```

javafx.application.Application

- init() - инициализирует приложение до его запуска. Метод не должен использоваться для создания графического интерфейса или отдельных его частей.
- start(Stage stage) - здесь определяется графический интерфейс.
- stop() - вызывается после закрытия приложения

Потоки

- Ни метод init, ни конструктор класса, который наследуется от Application, не подходит и не должен использоваться для создания графического интерфейса. Потому что и метод init, и конструктор запускаются в потоке, который называется потоком запуска или launcher thread.
- А создание и изменение графического интерфейса должно производиться в потоке приложения или application thread. Именно в таком потоке и запускается метод start (и метод stop).

Потоки выполнения

```
public static void main(String[] args) {
      System.out.println("Метод main()" + " " +
            Thread.currentThread().getName());
      Application.launch(args);
   }
```

```
@Override
  public void init() throws Exception {
    System.out.println("Метод init()" + " " +
        Thread.currentThread().getName());
  }


@Override
  public void stop() throws Exception {
    System.out.println("Метод stop()" + " " +
        Thread.currentThread().getName());
  }
```

```
@Override
    public void start(Stage stage) throws Exception {
        System.out.println("Метод start()" + " " +
                Thread.currentThread().getName());

        Label lb = new Label("Hello!!!");
        Pane root= new Pane(lb);
        Scene scene = new Scene (root,300,250);
        stage.setScene(scene);
        stage.setTitle("javaFX application");
        stage.show();
}
```

Структура приложения

- Каждое JavaFX приложение состоит из иерархии нескольких основных компонентов:
  - Stage (окна или подмостки)
  - Scene (сцены)
  - Node (узлы)

Stage

- Stage – Window – Object

- Stage()
- Stage(StageStyle s)

- В JavaFX application создается первичное окно, которое передается в метод start()

Окно по нажатию кнопки

```
Button bt = new Button(“Создать окно”);
root.setCenter(bt);
bt.setOnAction(event-> {
        newWind();
});
// добавляем метод  создания окна newWind()
public void newWind() {
        Stage wind = new Stage();
        wind.setTitle(“Новое окно”);
        wind.show();
}
```

Стили

StageStyle.DECORATED

StageStyle.UNDECORATED

StageStyle.TRANSPARENT

StageStyle.UTILITY

```
Stage w = new Stage(StageStyle.DECORATED);
```

```
Stage w = new Stage();
w.initStyle(StageStyle.UTILITY);
```

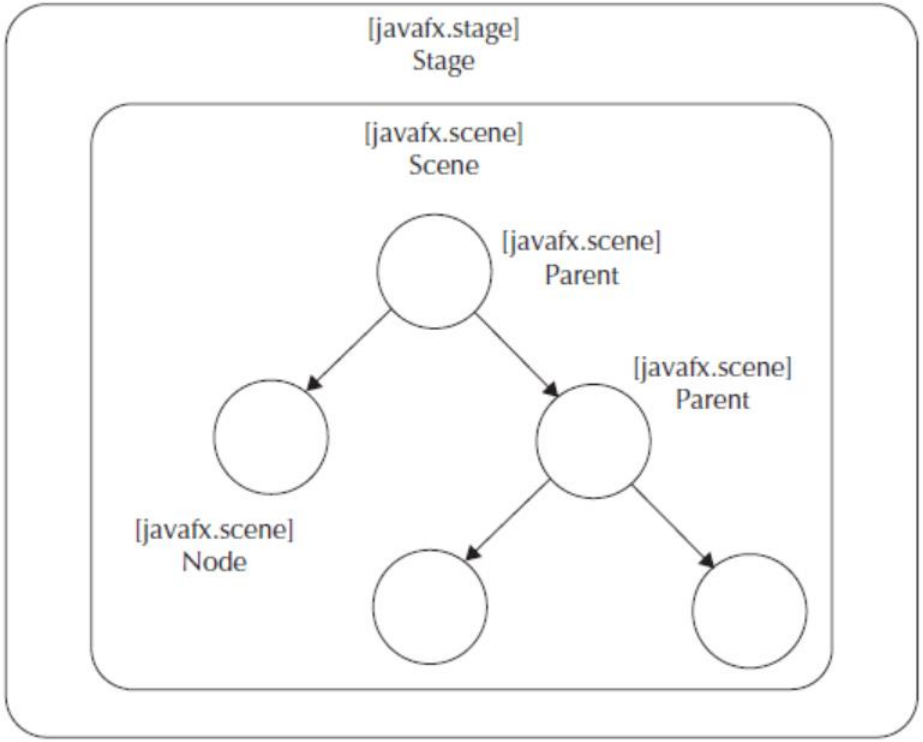
Управление

https://openjfx.io/javadoc/18/javafx.graphics/javafx/stage/Stage.html

По умолчанию открывается документация для JavaFX23

Scene

- javafx.scene.Parent
- javafx.scene.Node



- https://openjfx.io/javadoc/18/javafx.graphics/javafx/scene/package-summary.html

Scene

Scene(Parent root)

Scene(Parent root, double width, double height)

Scene(Parent root, Paint fill)

Scene(Parent root, double width, double height, Paint fill)

Организация курса

- Лекции по основам дизайна UI/UX
- Лекции и лабораторные занятия по JavaFX
- Выступления по темам дизайна UI/UX, инструментам и некоторым особенностям создания интерфейса в JavaFX
- Выполнение заданий на JavaFX
- Выполнение разработки в Figma