**Проект №3 — Тест-дизайн**

Привет, участник Школы 21! В этом проекте мы познакомим тебя с тест-дизайном, разберём, что это такое, зачем он нужен, какие существуют техники тест-дизайна и как ими пользоваться. Конечно, не обойдётся без практики.😉

Поехали!

**Instructions**

Напоминаем, что все отчёты по результатам выполнения заданий тебе нужно оформлять в файлах с расширением .md. Все созданные отчёты и файлы тебе нужно будет загрузить в папку src/ в корне проекта (обязательно в ветку *develop*). Если они уже созданы, то пересоздавать или удалять их не нужно (просто отредактируй этот файл).

**Contents**

1. [Chapter I](https://edu.21-school.ru/project/62665/task#chapter-i)  
   1.1. [Общая инструкция](https://edu.21-school.ru/project/62665/task#%D0%BE%D0%B1%D1%89%D0%B0%D1%8F-%D0%B8%D0%BD%D1%81%D1%82%D1%80%D1%83%D0%BA%D1%86%D0%B8%D1%8F)
2. [Chapter II](https://edu.21-school.ru/project/62665/task#chapter-ii)  
   2.1. [Тест-дизайн](https://edu.21-school.ru/project/62665/task#%D1%82%D0%B5%D1%81%D1%82-%D0%B4%D0%B8%D0%B7%D0%B0%D0%B9%D0%BD)  
   2.2. [Классы эквивалентности](https://edu.21-school.ru/project/62665/task#%D0%BA%D0%BB%D0%B0%D1%81%D1%81%D1%8B-%D1%8D%D0%BA%D0%B2%D0%B8%D0%B2%D0%B0%D0%BB%D0%B5%D0%BD%D1%82%D0%BD%D0%BE%D1%81%D1%82%D0%B8)  
   2.3. [Граничные значения](https://edu.21-school.ru/project/62665/task#%D0%B3%D1%80%D0%B0%D0%BD%D0%B8%D1%87%D0%BD%D1%8B%D0%B5-%D0%B7%D0%BD%D0%B0%D1%87%D0%B5%D0%BD%D0%B8%D1%8F)  
   2.4. [Задание №1](https://edu.21-school.ru/project/62665/task#%D0%B7%D0%B0%D0%B4%D0%B0%D0%BD%D0%B8%D0%B5-1-%D0%BE%D0%BF%D1%82%D0%B8%D0%BC%D0%B0%D0%BB%D1%8C%D0%BD%D0%BE%D0%B5-%D1%82%D0%B5%D1%81%D1%82%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5-%D0%BF%D0%BE%D0%BB%D1%8F-%D0%B2%D0%B2%D0%BE%D0%B4%D0%B0)
3. [Chapter III](https://edu.21-school.ru/project/62665/task#chapter-iii)  
   3.1. [Сценарии использования](https://edu.21-school.ru/project/62665/task#%D1%81%D1%86%D0%B5%D0%BD%D0%B0%D1%80%D0%B8%D0%B8-%D0%B8%D1%81%D0%BF%D0%BE%D0%BB%D1%8C%D0%B7%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D1%8F)  
   3.2. [Задание №2](https://edu.21-school.ru/project/62665/task#%D0%B7%D0%B0%D0%B4%D0%B0%D0%BD%D0%B8%D0%B5-2-%D1%81%D0%BE%D1%81%D1%82%D0%B0%D0%B2%D0%BB%D0%B5%D0%BD%D0%B8%D0%B5-%D1%81%D1%86%D0%B5%D0%BD%D0%B0%D1%80%D0%B8%D0%B5%D0%B2-%D0%B8%D1%81%D0%BF%D0%BE%D0%BB%D1%8C%D0%B7%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D1%8F)  
   3.3. [Интуитивное тестирование](https://edu.21-school.ru/project/62665/task#%D0%B8%D0%BD%D1%82%D1%83%D0%B8%D1%82%D0%B8%D0%B2%D0%BD%D0%BE%D0%B5-%D1%82%D0%B5%D1%81%D1%82%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5)  
   3.4. [Попарное тестирование](https://edu.21-school.ru/project/62665/task#%D0%BF%D0%BE%D0%BF%D0%B0%D1%80%D0%BD%D0%BE%D0%B5-%D1%82%D0%B5%D1%81%D1%82%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5)  
   3.5. [Задание №3](https://edu.21-school.ru/project/62665/task#%D0%B7%D0%B0%D0%B4%D0%B0%D0%BD%D0%B8%D0%B5-3-%D0%BF%D0%BE%D0%BF%D0%B0%D1%80%D0%BD%D0%BE%D0%B5-%D1%82%D0%B5%D1%81%D1%82%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5)

**Chapter I**

**Общая инструкция**

Методология Школы 21 может быть не похожа на тот образовательный опыт, который случался с тобой ранее. Её отличает высокий уровень автономии: у тебя есть задача, ты должен её выполнить. По большей части тебе нужно будет самому добывать знания для её решения. Второй важный момент — это peer-to-peer обучение. В образовательном процессе нет менторов и экспертов, перед которыми ты защищаешь свой результат. Ты это делаешь перед такими же учащимися, как и ты сам. У них есть чек-лист, который поможет им качественно выполнить приемку вашей работы.

Роль Школы 21 заключается в том, чтобы обеспечить через последовательность заданий и оптимальный уровень поддержки такую траекторию обучения, при которой ты не только освоишь hard skills, но и научишься самообучаться.

* Не доверяй слухам и предположениям о том, как должно быть оформлено ваше решение. Этот документ является единственным источником, к которому стоит обращаться по большинству вопросов;
* твое решение будет оцениваться другими учащимися;
* подлежат оцениванию только те файлы, которые ты выложил в GIT (ветка develop, папка src);
* в твоей папке не должно быть лишних файлов — только те, что были указаны в задании;
* не забывай, что у вас есть доступ к интернету и поисковым системам;
* обсуждение заданий можно вести и в Rocket.Chat;
* будь внимателен к примерам, указанным в этом документе — они могут иметь важные детали, которые не были оговорены другим способом;
* и да пребудет с тобой Сила!

**Chapter II**

**Тест-дизайн**

**Тест-дизайн** — это этап процесса тестирования ПО, на котором проектируются и создаются тест-кейсы в соответствии с определёнными ранее критериями качества и целями тестирования.

**Тест-дизайнер** — это сотрудник, в чьи обязанности входит создание тест-кейсов, обеспечивающих оптимальное тестовое покрытие тестируемого продукта. На самом деле, для любого тестировщика важно знать и использовать техники тест-дизайна, так как без них количество тест-кейсов возрастает в разы, а процент проверяемости приложения, наоборот, сильно падает.

Что же это за техники такие?🧐 Среди основных можно выделить следующие:

* Классы эквивалентности;
* Граничные значения;
* Сценарии использования;
* Попарное тестирование;
* Интуитивное тестирование;

**Классы эквивалентности**

**Класс эквивалентности** — одно или несколько значений ввода, к которым программное обеспечение применяет одинаковую логику.

Техника анализа классов эквивалентности заключается в том, что мы разделяем функционал (часто разделяется множество возможных входных значений) на группы эквивалентных по своему влиянию на систему частей с целью дальнейшего сокращения количества тестов. Такое разделение помогает убедиться в правильном функционировании целой системы — одного класса эквивалентности, — проверив только один элемент этой группы.

Допустим, у нас есть поле ввода возраста. На сайте указано, что допустимые значения — целые числа от 1 до 100. Как проверить все возможные случаи? И сколько нам нужно это проверять, если поле позволяет вводить любые символы и значения? Если так подумать, то проверять мы можем бесконечное количество раз всё новые и новые значения. Например, проверить 1, потом 1.1, 1.01 и так далее. Понятно, что если 1.1 не сработает, вряд ли есть смысл проверять 1.01. Для этого и существуют классы эквивалентности. Например, для нашего примера с полем ввода возраста можно составить следующие классы эквивалентности:

1. числа меньше 1 (в том числе отрицательные числа);
2. целые числа от 1 до 100;
3. нецелые числа от 1 до 100;
4. числа больше 100;
5. специальные символы (# @ + — / \_ : ; “ ‘ и т.д.);
6. буквы.

Среди вышеперечисленных классов эквивалентности только один проверяет допустимые значения — это класс №2 (целые числа от 1 до 100). На тесты, относящиеся к другим классам, сайт должен уведомить пользователя, что было введено неверное значение в поле ввода возраста.

**Граничные значения**

**Граничные значения** — значения диапазона входных данных, при которых меняется поведение приложения. Другое определение: граничные значения — это значения, в которых один класс эквивалентности переходит в другой.

Подробно изучи этот вопрос самостоятельно и переходи к заданию!😉

**Задание №1. Оптимальное тестирование поля ввода**

Используя полученные знания (классы эквивалентности и граничные значения), опиши все *минимально необходимые* тест-кейсы для проверки поля ввода возраста. Условия те же — допустимыми значениями являются целые числа от 1 до 100 включительно. Запиши полученные тест-кейсы в документ exercise1.md

**Chapter III**

**Сценарии использования**

**Сценарии использования** (Use Cases, часто — юзкейсы) описывают сценарии взаимодействия двух и более участников (как правило – пользователя и системы). Пользователем может выступать как человек, так и другая система. Для тестировщиков Use Case являются отличной базой для формирования тестовых сценариев (тест-кейсов), так как они описывают, в каком контексте должно производиться каждое действие пользователя. Use Case, по умолчанию, являются проверяемыми требованиями, так как в них всегда указана цель, которую нужно достигнуть, и шаги, которые надо для этого воспроизвести.

Важно различать тестовые сценарии и сценарии использования. Приведём также пример сценария использования:

- Сценарий: работа с модулем аудита

- Как: пользователь системы с правом работы с модулем аудита

- Чтобы: просмотреть статистику в модуле аудита

1. Дано: я нахожусь на странице модуля аудита

2. Чтобы просмотреть статистику в модуле аудита:

- Мне нужно заполнить необходимую информацию в шторке для настройки параметров

- Если шторка скрыта и я не вижу раздела для настройки параметров, мне нужно кликнуть на кнопку “Настройка параметров” в левой части страницы

- В параметре "Начало временного диапазона", мне нужно выбирать дату, которая послужит отправной точкой для расчета статистических показателей

- В параметре "Конец временного диапазона" мне нужно выбрать дату, до которой будет расчитана статистика

3. Когда все параметры будут настроены:

- Я увижу статистику в модуле аудита

**Задание №2. Составление сценариев использования**

Перейди на страницу <https://sbermarket.ru/>. Открой модальное окно авторизации. В файле exercise2.md (создай этот файл!) опиши не менее семи сценариев использования этого модального окна.

К примеру, темой одного из сценариев будет: "Получение выгодных предложений" или "Вход через VK".

**Интуитивное тестирование**

**Интуитивное тестирование** — вид тестирования, который выполняется без подготовки к тестам, без определения ожидаемых результатов, проектирования тестовых сценариев. Это неформальное, импровизационное тестирование. Оно не требует никакой документации, планирования, процессов, которых следует придерживаться в выполнении. Также на данный вид тестирования не пишутся тест-кейсы, что в свою очередь может вызвать определенные затруднения в попытках воспроизвести дефект в системе. Возможно, именно так себе и представляют работу тестировщика, но на самом деле такой вид тестирования имеет смысл. Интуитивное тестирование зачастую может дать сходу больше результатов, чем тестирование по заранее определенным сценариям, так как тестировщик может проверять нестандартные случаи.

**Попарное тестирование**

**Попарное тестирование** — техника тест-дизайна, которая обеспечивает полное тестовое покрытие. Основная идея данной техники заключается в том, что при большом количестве параметров, проверить все возможные их комбинации слишком времязатратно. При попарном тестировании предполагается, что если в процессе тестирования были проверены все комбинации для каждой отдельно взятой пары параметров, то полученное покрытие эквивалентно таковому при полном переборе значений всех возможных комбинаций всех параметров. Проще говоря, можно проверять не каждое значение или каждое поле по отдельности, а комбинировать их и проверять парами!

P.S.: Обязательно найди в интернете информацию про тестовое покрытие!🙂

**Задание №3. Попарное тестирование**

Перейди на страницу авторизации <https://www.saucedemo.com/>. Необходимо проверить два поля — "Username" и "Password", используя технику попарного тестирования.

Создай файл exercise3.md, для каждого поля опиши значения, которые они могут принимать (пустое, верное значение логина и так далее). Потом сопоставь их друг с другом и вместо проверки каждого значения, составь тест-кейсы, которые проверяют сразу два значения за раз. Сначала может показаться, что это очень просто, но чем больше становится в форме полей, которые нужно проверять, тем больше становится самих тестовых случаев, и проверять их по отдельности — не самое оптимальное решение.

**Double-check**

Перед загрузкой выполненного проекта в репозиторий перепроверь наличие всех необходимых файлов, которые требовалось создать во время его выполнения:

exercise1.md

exercise2.md

exercise3.md

💡 [Нажми здесь](https://forms.gle/Gn1uLREZzXbAwy6LA) **чтобы отправить обратную связь по проекту**.