День 4. Прототип проекта

**Прототипи́рование программного обеспечения** (от англ. prototyping) — этап разработки программного обеспечения (ПО), процесс создания **прототи́па программы** — макета (черновой, пробной версии) программы, обычно — с целью проверки пригодности предлагаемых для применения концепций, архитектурных и/или технологических решений, а также для представления программы заказчику на ранних стадиях процесса разработки.

Прототип позволяет также получить обратную связь от будущих пользователей, причем, именно тогда, когда это наиболее необходимо: в начале проекта еще есть возможность исправить ошибки проектирования практически без потерь.

Процесс создания прототипа обычно состоит из шагов:

1. Определение начальных требований
2. Разработка первого варианта прототипа, который содержит только пользовательский интерфейс системы
3. Изучение прототипа заказчиком и конечными пользователями, получение обратной связи о необходимых изменениях и дополнениях
4. Переработка и улучшение прототипа: с учетом полученных замечаний и предложений изменяются как спецификации так и прототип, после этого шаги 3 и 4 могут повторяться.

Прототипирование имеет множество различных вариантов. Тем не менее, все методы в какой-то степени основаны на двух основных типах.

**Быстрое прототипирование**

При быстром прототипировании ([англ.](https://ru.wikipedia.org/wiki/%D0%90%D0%BD%D0%B3%D0%BB%D0%B8%D0%B9%D1%81%D0%BA%D0%B8%D0%B9_%D1%8F%D0%B7%D1%8B%D0%BA) *rapid prototyping* или *throwaway prototyping*) предполагается, что создаётся макет, который на каком-то этапе будет оставлен («выброшен») и не станет частью готовой системы.

Основное преимущество такого подхода — в скорости: в ответ на свои требования заказчик почти сразу получает прототип интерфейса, и сразу может уточнить требования, до того как начато написание рабочего кода системы. Стоимость изменения требований на этом этапе очень низкая, поскольку нет кода, который нужно было бы переписывать.

Очень важно, чтобы такое прототипирование было выполнено в кратчайшие сроки, поскольку в данном случае тратятся время и ресурсы на код, который не будет в дальнейшем использован.

Быстрое прототипирование не обязательно выполняется в рамках той же платформы и тех же технологий, что и разрабатываемая система. Для прототипа [графического интерфейса пользователя](https://ru.wikipedia.org/wiki/%D0%93%D1%80%D0%B0%D1%84%D0%B8%D1%87%D0%B5%D1%81%D0%BA%D0%B8%D0%B9_%D0%B8%D0%BD%D1%82%D0%B5%D1%80%D1%84%D0%B5%D0%B9%D1%81_%D0%BF%D0%BE%D0%BB%D1%8C%D0%B7%D0%BE%D0%B2%D0%B0%D1%82%D0%B5%D0%BB%D1%8F) (GUI) могут использоваться как стандартные [HTML](https://ru.wikipedia.org/wiki/HTML)-страницы, либо прототип может подготавливаться в программе, специально предназначенной для создания макетов (например: [Axure RP](https://ru.wikipedia.org/wiki/Axure_RP" \o "Axure RP), [Microsoft Expression Blend](https://ru.wikipedia.org/wiki/Microsoft_Expression_Blend" \o "Microsoft Expression Blend) и др.).

**Эволюционное прототипирование**

*Эволюционное прототипирование* ([англ.](https://ru.wikipedia.org/wiki/%D0%90%D0%BD%D0%B3%D0%BB%D0%B8%D0%B9%D1%81%D0%BA%D0%B8%D0%B9_%D1%8F%D0%B7%D1%8B%D0%BA) *evolutionary prototyping*) ставит своей целью последовательно создавать макеты системы, которые будут все ближе и ближе к реальному продукту.

Такой подход имеет то преимущество, что на каждом шаге мы располагаем работающей системой, пусть и не обладающей всей нужной функциональностью, но улучшающейся с каждой итерацией. При этом, не тратятся ресурсы на код, который будет «выброшен».

Эволюционный подход к прототипированию может быть выбран, исходя из предположения, что все необходимые требования к моменту начала разработки неизвестны, и будут определяться по мере создания программы; тогда на каждом этапе мы реализуем лишь те требования, которые известны и ясны. Иногда при этом разработчики сосредотачиваются на работе только над теми модулями системы, требования на которые уже определены.

В некоторых случаях, когда речь идет о продукте под определенную незанятую нишу, пользователи начинают использовать систему еще до того, как она полностью дописана, в ожидании готовой системы, поскольку «недописанная система — это лучше, чем её полное отсутствие».

Основными преимуществами прототипирования являются сокращение времени и стоимости разработки за счёт того, что оценка прототипа позволяет на более ранних стадиях обнаружить недостаточность или несоответствие требований. Чем позднее проводятся изменения в спецификации, тем они дороже, поэтому уточнение «чего же пользователи/заказчики хотят *на самом деле*» на ранних стадиях разработки снижает общую стоимость.

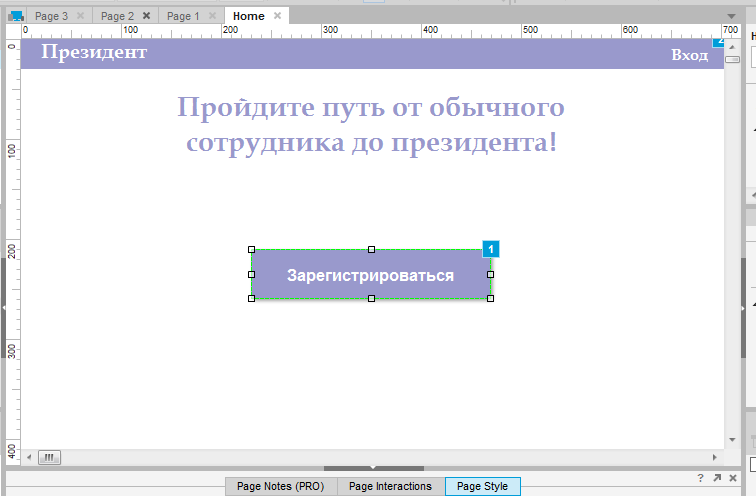
Психологически важную роль играет также вовлечение заказчика в процесс разработки. Работа с прототипом позволяет будущим пользователям увидеть, как будет выглядеть будущая программа, и повлиять на её поведение, что уменьшает расхождения в представлении о программе между разработчиками и пользователями. Снижается и эффект почти неизбежного отторжения новой системы при внедрении, особенно когда она внедряется на место ранее использовавшейся.

Тем не менее, использование прототипирования создаёт ряд дополнительных рисков.

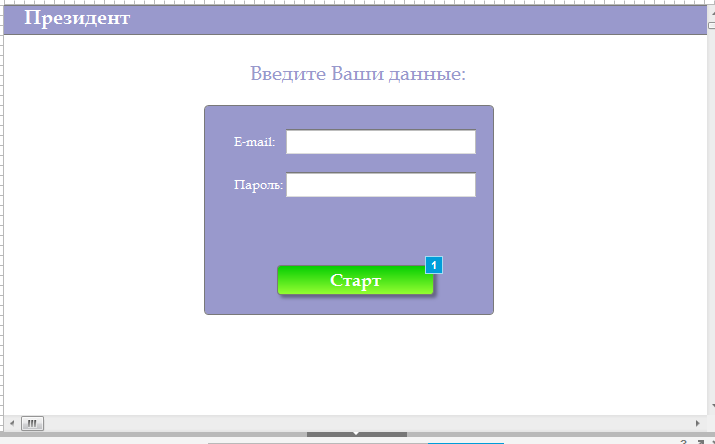
* Риск недостаточного анализа. Концентрация усилий на ограниченном прототипе может отвлекать разработчиков от надлежащего анализа требований на полную систему.
* Риск смешения прототипа и готовой системы в представлении пользователей. Пользователи могут подумать, что прототип, который в действительности предполагается «выбросить», и есть основа будущей системы. Это может привести к двум противоположным по сути негативным эффектам. Во-первых, пользователи могут ожидать от прототипа более точного поведения и, не обнаружив его, разочароваться в возможностях разработчиков. Во-вторых, наличие быстро созданного прототипа может создать впечатление того, что «почти вся работа уже сделана», что может стимулировать попытки неоправданно сократить время и/или бюджет разработки.
* Риск потери времени на создание прототипа. Ключевое свойство прототипа — то, что он создается быстро. Если разработчики тратят время на создание слишком сложного и функционального прототипа, они теряют преимущества от применения прототипирования вообще.

**Прототип проекта**

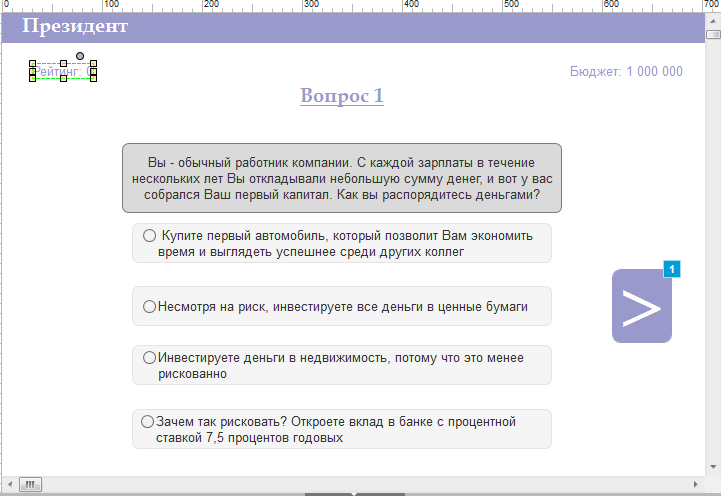
1. Домашняя страница сайта



1. Страница авторизации



1. Тест



1. Конец игры.

