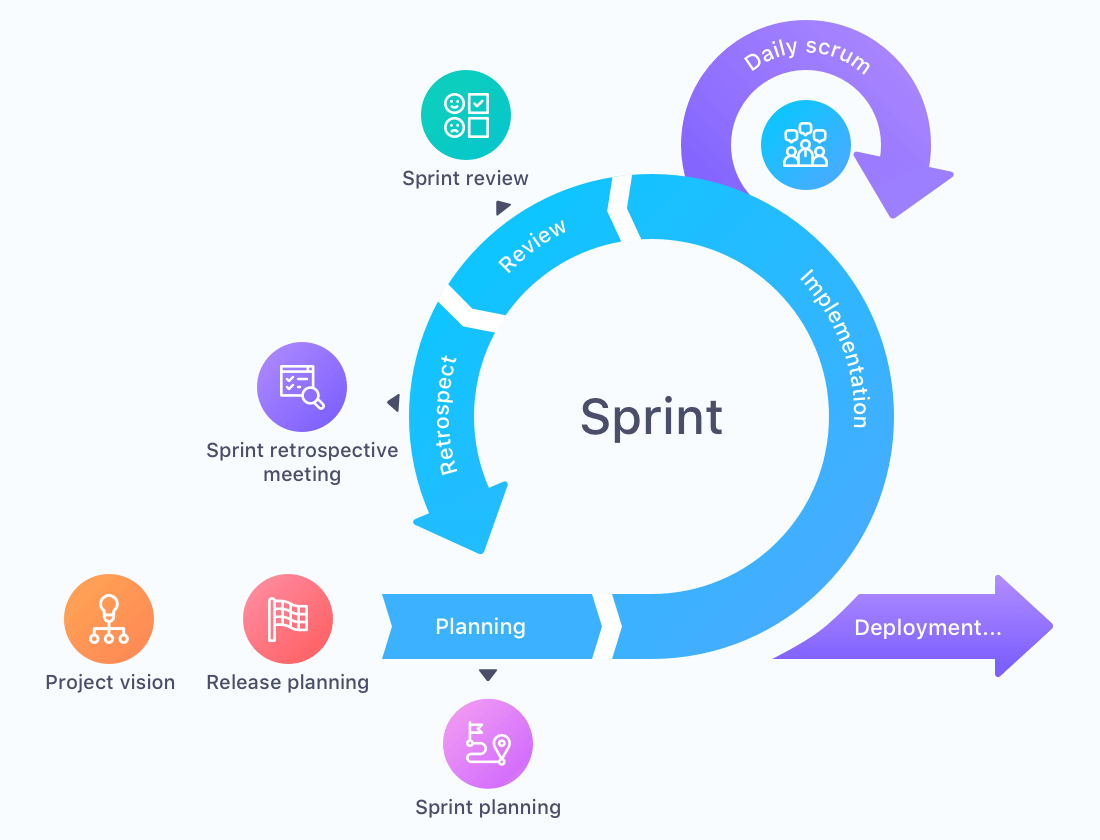
**SCRUM**

Scrum — это фреймворк управления проектами, который предназначен для того, чтобы обеспечить эффективность и гибкость управления проектами в условиях неопределённости и быстро меняющихся требований. Данный метод реализуется в рамках Agile-методологии.

Scrum это не свод правил, а лишь фреймворк, который описывает принципы, как может работать команда в тех случаях, когда ей часто нужно проверять делает ли она то, что нужно.

Лучше всего описывает работу по scrum, это изображение:

Здесь описаны все мероприятия, которые и создают гибкую среду разработки. Коротко о том, что происходит при разработке программного продукта: проект разбивают на спринты – промежутки времени (обычно 2-3 недели), в которые команда разрабатывает продукт. В конце каждого спринта команда показывает готовую часть продукта – либо его неидеальную версию, которой уже можно пользоваться, либо дополнительный функционал. По другому говоря, результат спринта называется Increment. Increment – это конкретная ступенька к достижению цели программного продукта, пазл программного продукта.

На примере нашего проекта: в конце первого спринта наша команда делает весь основной функционал для управления рестораном (есть возможность регистрации, входа, брони столиков и управления заказами). Во время второго спринта команда делает, например, возможность обращения в тех. Поддержку и так далее. При этом рабочая версия продукта сделана уже в конце первого спринта.

Если говорить подробнее, то спринт делится на следующие этапы: планирование, анализ, выполнение, тестирование и релиз.

**Планирование**. С этого этапа начинается каждый спринт. Здесь обсуждают требования заказчика, формируют бэклог продукта и затем приоритизируют задачи из бэклога и самые важные размещают в его верху.

**Анализ**. Участники команды выбирают задачи для спринта из самого верха спринта (т.е. формируется бэклог спринта). Затем определяют, какие ресурсы понадобятся для выполнения задач, и распределяют эти задачи между собой. Здесь же и формируют критерии успешного завершения каждой задачи, т.е. параметры, по которым будет понятно, что задача выполнена хорошо.

**Выполнение**. На этом этапе участники команды просто выполняют поставленные задачи. Scrum команда собирается каждый день на 15-ти минутный scrum-митинг, цель которого является понять, всё ли идёт по плану и, что нужно для преодолений препятствий. Каждый участник scrum-команды на этом митинге отвечает на 3 вопроса:

* что он сделал с прошлой встречи;
* что планирует делать сегодня;
* что ему в этом мешает;

**Тестирование**. На этом этапе проверяют работоспособность продукта: всё ли запускается и функционирует верно. Всё это нужно для быстрого реагирования на обнаруженные проблемы и вносить изменения вовремя.

**Релиз**. Конечно же результат каждого спринта обсуждают с заказчиком и, если его что-то не устраивает, команда обсуждает, как будет дорабатывать проект, и возвращается к предыдущим этапам. Если же замечаний нет, то команда приступает к следующей итерации. Требования могут меняться в ходе разработки, что как раз и подразумевает методология.

**Ретроспектива**. Является заключающим мероприятием в спринте. Здесь участники команды обсуждают, что прошло хорошо во время спринта и какие проблемы возникли во время него. Результатом этапа являются предположения, которые сбили команду с пути, команда исследует их предположение.

Если же во время спринта что-то идёт не так – например, кто-то из членов команды понимает, что не укладывается в сроки, он сообщает об этом владельцу продукта, и тот перераспределяет время и задачи. Если же команда заканчивает всё досрочно, то в рамки спринта могут добавляться дополнительные задачи из бэклога.

**Роли в scrum-команде**. Что вообще такое scrum-команда? Scrum-команда – это основная единица scrum, небольшая команда людей (до 10 человек). В команде есть 3 ключевые роли: команда разработчиков (developers), скрам-мастер (scrum master) и владелец продукта (product owner). Говоря про каждую роль отдельно:

Разработчики – это люди, которые причастны к созданию любого аспекта готового к использованию Increment в каждом спринте, иначе – это люди, которые создают продукт. К ним относятся разработчики, тестировщики, дизайнеры и другие специалисты.

Скрам-мастер – человек, который несёт ответственность за применение Scrum в соответствии с Руководством по Scrum. Он направляет команду и решает проблемы, замедляющий процессы. Это не руководитель команды, а скорее наставник, задача которого организовать работу так, чтобы каждый участник команды понимал потребности заказчика и мог предлагать свои идеи. Он также организует общение заказчика и команды на мероприятиях.

Владелец проекта – человек, отвечающий за то, чтобы у команды была чёткая цель и понимание того, какой продукт нужно создать. Product owner сотрудничает с заказчиком и другими заинтересованными сторонами. В некоторых проектах заказчик и есть product owner. Также владелец формирует и приоритизирует список требований к продукту – бэклог – и следит за тем, чтобы команда работала в нужном направлении.

Все эти специалисты тесно взаимодействуют друг с другом. Product owner задаёт направление проекта и приоритеты, developers реализуют задачи, а scrum-master отвечает за то, чтобы процесс разработки шёл гладко.

Принципы scrum. Цель scrum – создать желаемый продукт вовремя с минимальными затратами ресурсов. Чтобы этого достичь, команда опирается на следующие принципы:

* Кросс-функциональность. Все участники имеют различные специализации. Это позволяет эффективно работать над продуктом, не привлекая никого извне.
* Постоянное улучшение. Команда регулярно анализирует свою работу на scrum-митингах, что помогает выявлять и решать проблемы во время разработки. Таким образом команда постоянно совершенствует продукт.
* Прозрачность. У всех в команде есть чёткое представление о целях и задачах в проекте, чему опять же способствуют scrum-митинги.
* Фокус на ценностях. На каждом спринте команда фокусируется только на самых приоритетных задачах из бэклога.

Внедрение scrum может показаться сложным на первых порах из-за непонятных интерпретация и множества нюансов. Однако, опираясь на фундаментальные принципы руководства по Scrum и постоянно обучаясь, можно значительно улучшить этот процесс.

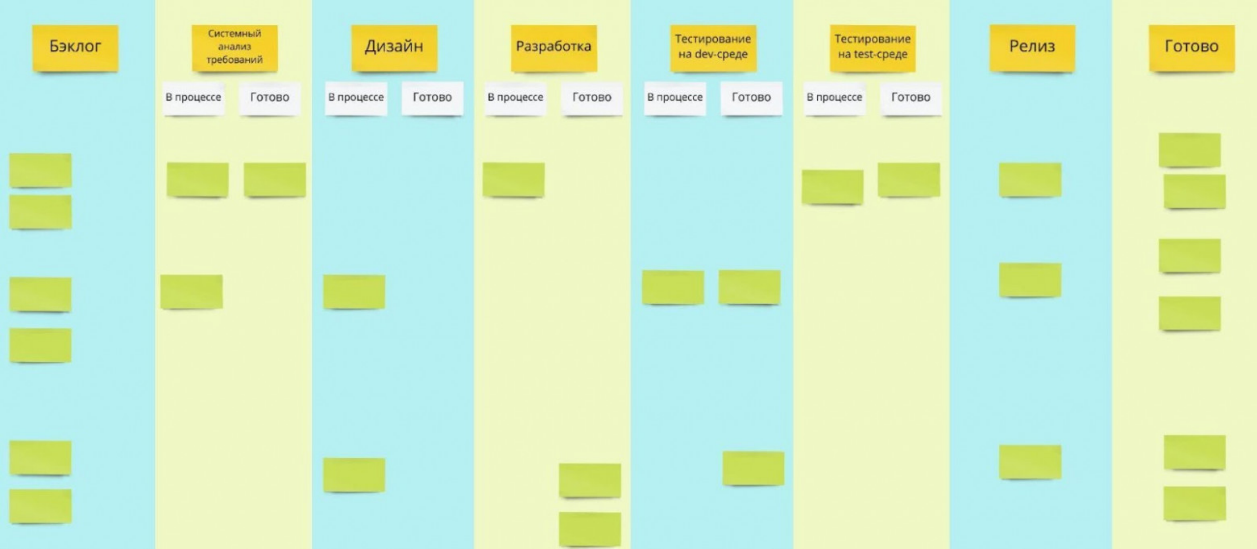
Kanban

Kanban – фреймворк для улучшения процессов разработки, часть agile-философии. Построен на практический той же основе, что и scrum – “Манифест гибкой разработки ПО”.

**Целью Kanban** можно считать такой же, как и у scrum – получить готовый качественный продукт вовремя, с минимальными затратами. Чтобы добиться этого, в Kanban есть несколько ключевых практик. Первое – визуализация процесса разработки.

**Доски Kanban**. Для визуализации рабочего процесса используются доски Kanban. Доступ к таким доскам есть у каждого члена команды в любой момент времени, так что он видит, на каком этапе находится задача.

Доска может быть как виртуальной, так и реальной. Kanban доска подстраивается под любой процесс и применяется в любой области. Пример, как бы выглядела доска для какого-нибудь IT проекта:



Как мы видим, есть некоторые этапы. Каждый такой этап может находиться в различных состояниях. Определённые состояния сигнализируют о том, что работа над этапом ещё не закончена, либо о том, что можно перейти к следующему этапу.

Имена столбцов меняются в зависимости от проекта, однако важно сохранять их последовательность – это одна из ключевых ценностей Kanban, которую называют потоком. Одна из важных практик Kanban-а является управление потоком работы. Чтобы задачи завершались быстро, равномерно и предсказуемо, а команда фокусировалась на результате, в первую очередь следят за потоком задач, а не за людьми. Чтобы управлять этим потоком, в Kanban есть подобие scrum-митингов, где планируется работа на день и обновляется доска. Синхронизация происходит справа налево. Команда в первую очередь берётся за задачи, которые близки к колонке “Готово” и задаёт вопрос: “А что мы сделаем, чтобы передвинуть эту задачу в следующую колонку?”. Так и разбирается каждая задача и составляется план работы с ней.

Другая важная практика в Kanban – **ограничение количества задач** (WIP-лимит). Необходимо ограничивать количество задач, которые одновременно берутся в работу, это помогает увеличить скорость их выполнения и создать прогнозируемую систему, чтобы заказчики знали сроки работы.

Зачем вообще нужны лимиты? Пример: если в команде всего 2 разработчика, а одновременно на этапе разработки выполняется 10 задач, возникают простои, так как в один момент времени один разработчик способен выполнять только одну, максимум две задачи, переключаясь между ними. Если же задач больше, то нельзя будет сфокусироваться на одной, разработчик постоянно будет переключаться между ними, теряя эффективность работы.

Лимит можно поставить сразу в начале работы, он будет вычисляться по следующей формуле: Лимит = Человек в команде – 1. Конечно лимиты могут меняться по ходу работы, так сказать они будут определятся эволюционно и эмпирически: мы смотрим, как движется работа, какие есть возможности на каждом этапе и исходя из этого ставим лимиты.

Если же лимиты отличаются в зависимости от этапа задач, например, команда способна выполнять 5 задач за неделю, но при этом тестировать всего 2 задачи, то здесь лимит будет равняться тому самому узкому месту – 2 задачи.

**Основные принципы Kanban**. Так же, как и в scrum, чтобы достичь нужного результата, команда должна опираться на некоторые принципы:

* Уважать и использовать то, что есть сейчас: имеющиеся роли, обязанности и должностные инструкции.
* Постоянно оптимизировать и совершенствовать процесс разработки, но не допускать слишком резких перемен.
* Поощрять в команде стремление к лидерству.

Сравнение SCRUM и Kanban

Так как фреймворки в некоторых местах бывают похожи, следует провести их сравнение.

|  |  |  |
| --- | --- | --- |
|  | SCRUM | Kanban |
| Итерации | Фиксированные спринты | Непрерывный поток |
| Гибкость | Изменения только между спринтами | Изменения в любое время |
| Роли | Строгое разделение | Нет обязательных ролей |
| Команда | Могут работать узкопрофильные команды | Только кросс-функциональные команды |
| Подходит для | Проекты с меняющимися требованиями | Поддержка, операционные задачи |

**Заключение**. Scrum идеален для проектов с чёткими целями, но гибкими требованиями (разработка ПО, стартапы), Kanban же в свою очередь подходит для задач с нерегулярным потоком (поддержка, IT-операции).