СЛОВАРЬ IT-ТЕРМИНОВ

skypro

Для тех, кто на «вы» с IT

ПРОДУКТ, ПРОГРАММНЫЙ ПРОДУКТ ИЛИ ПО

Система или алгоритм, которые создает команда айтишников.
Например, это сайт или приложение, которые разрабатывают для пользователей.

AGILE

Гибкий метод разработки, набор принципов о том, как разрабатывать продукт.
Он появился в 2001 году, а до этого команды планировали разработку более жестко.

Например, Ася и Света одновременно купили квартиры. Ася наняла строителей и дизайнера, согласовала сроки выполнения проекта и отправила исполнителей делать ремонт. Ася не въезжает в квартиру, пока не закончат работы. В итоге еще год живет на съемной квартире и ждет не дождется, когда переедет.

А Света заехала в квартиру сразу. Пока она живет на матрасе в спальне, строители и дизайнеры оформляют ей гостиную. Когда закончат с гостиной, то переберутся в спальню, а Света — в гостиную. В итоге Света с первого дня живет в квартире и делает все постепенно. Причем ремонт делает минимальный, а декором займется, когда закончит основные работы. Света делает ремонт по Agile.

> **Основные принципы метода:**
>
> — люди и взаимодействие важнее процессов и инструментов;
> — работающий продукт важнее исчерпывающей документации;
> — сотрудничество с заказчиком важнее согласования условий контракта;
> — готовность к изменениям важнее следования первоначальному плану.

Если ты сменишь профессию на айтишную, то, скорее всего, будешь работать по этим принципам. Управлять разработкой по Scrum или Kanban, делить работу на спринты, начинать день со Stand-up, а заканчивать спринт Retro и Demo.

SCRUM, ИЛИ СКРАМ

Метод управления разработкой, когда команда определяет набор задач на несколько недель и, пока не закончит, не меняет приоритеты.

Например, ты запланировал, что на ближайшие две недели станешь вегетарианцем. Тогда, пока две недели не кончатся, ты не будешь менять рацион, даже если он тебе не понравится.

KANBAN, ИЛИ КАНБАН

Метод управления разработкой, когда приоритеты и задачи меняются в любой момент, а жестких сроков для итога нет.

Например, при Kanban ты уже на третий день поймешь, что не можешь жить без яичницы на завтрак, и спокойно вернешь ее в рацион. Этот подход — более гибкий.

SPRINT, ИЛИ СПРИНТ

Отрезок времени, за который команда решает часть задач. Спринты длятся две-три недели, и по окончании срока задача точно должна быть готова — ее нельзя переносить на следующий спринт.

Например, ты пообещал племяннику, что через три недели сошьешь ему пиджак, как раз к первому сентября. Все три недели ты будешь искать выкройки, ткани, шить, звать племянника на примерку и в итоге отдашь ему готовый пиджак.

Только в IT работают в команде, поэтому и задачи на спринт добавляют в общий бэклог.

BACKLOG, ИЛИ БЭКЛОГ

Список задач, которые команда берет на весь период разработки продукта.

Например, ты составляешь такой же бэклог, когда пишешь цели на год. Ты написал 60 целей, но это не значит, что ты воплотишь их в ближайший месяц.

SPRINT BACKLOG, ИЛИ БЭКЛОГ СПРИНТА

Задачи на конкретный спринт.

Например, из целей на год ты решаешь взять первые пять на январь — это твой бэклог на спринт.

У айтишников так же, только работают они в команде, поэтому все задачи нужно обсуждать и отсеивать менее приоритетные.

GROOMING, ИЛИ ГРУМИНГ

«Стрижка» задач, которые пойдут в бэклог на спринт. Во время груминга команда оценивает, какие задачи уже неактуальны, а какие можно взять на следующие три недели.

Например, к августу ты понял, что не хочешь поступать в университет, чтобы получить второе высшее образование, и вычеркиваешь «Готовиться к вступительным экзаменам» из списка задач.

STAKEHOLDERS, ИЛИ СТЕЙКХОЛДЕРЫ

Люди и организации, которые влияют на итоговый продукт. Так называют всех, кто задействован в разработке продукта: команду айтишников, спонсоров, заказчиков, акционеров и менеджеров.

JUNIOR, MIDDLE И SENIOR

Айтишники делятся по уровням: junior, middle и senior.

> Junior, или джун, — это новичок в IT, который только окончил обучение, прошел стажировку или поработал на стартовой позиции несколько месяцев.
>
> Middle, или мидл, — более опытный специалист. Чаще всего — это джун с опытом работы от полугода до двух лет.
>
> Senior, или сеньор, — самый опытный айтишник, который руководит джунами и мидлами. Чтобы стать сеньором, джуну нужно 3–6 лет.

BACKEND, FRONTEND И FULL-STACK

А разработчики делятся по направлениям: backend, frontend и full-stack.

> Backend-разработчик пишет «внутреннюю» часть продукта. Разрабатывает все, что не видит и с чем не соприкасается пользователь, — «движок» сайта или приложения.
>
> Frontend-разработчик отвечает за то, каким продукт видит пользователь. Верстает, оживляет и адаптирует сайт или приложение и его элементы.
>
> Full-stack-разработчик делает и то и то. А еще поддерживает инфраструктуру продукта. Подключается к задачам backend- и frontend-разработчиков, управляет проектами и объединяет работу специалистов команды.

PLANNING POKER

Метод оценки задач из бэклога. Его применяют на груминге, чтобы определить, какие задачи можно взять в бэклог спринта и уложиться в срок. Задачу оценивают в рабочих часах, ставят оценку относительно самой маленькой и самой большой.

Например, ты оцениваешь рынок тату-мастеров и понимаешь, что на татуировку нужно 10 000 ₽ и три сеанса. Если найдешь лишние деньги в бюджете и несколько часов в расписании, то пойдешь бить тату в этом месяце. Если нет — отложишь на потом.

ФИЧА

Во время спринтов команда внедряет фичи.

Фича — новая функциональность в продукте. Без нее продукт и так работает хорошо, но фича приятно дополняет основной набор функций.

Например, ты и так выглядишь отлично, но решаешься набить татуировку. Татуировка — твоя новая фича. Чтобы разобраться, сколько денег и времени нужно на татуировку, нужен покер планирования.

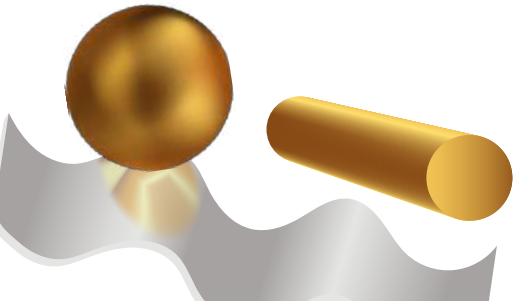

ДИАГРАММА ГАНТА

Это инструмент управления проектами, который помогает отслеживать, как задачи влияют друг на друга. Обычно это график, где в левой части описана задача, а в правой тянется полоса в количестве рабочих часов, которые исполнитель тратит на задачу. С помощью диаграммы проще следить за проектом и управлять прогрессом, визуализировать задачи на спринт.

STAND-UP MEETING, ИЛИ СТЕНДАП

Встреча в начале дня, на которой каждый член команды делится планами. Это ежедневный отчет перед командой, на котором все озвучивают проблемы и как их будут решать.

Ты проводишь такой стендап, когда садишься с друзьями в кафе и рассказываешь, что хочешь татуировку, но в городе нет мастера, который тебе нужен, и ты поедешь бить тату в соседний город через неделю.

DOD (DEFINITION OF DONE)

Это образ конечного результата. Перед выполнением задачи команда определяет, как именно поймет, что задача закончена, по каким показателям отследить прогресс.

В случае с татуировкой DOD — это готовая татуировка у тебя на руке после трех сеансов с мастером.

RETRO, ИЛИ РЕТРОСПЕКТИВА

Встреча в конце спринта, на которой команда обсуждает результаты спринта. Цель встречи — поделиться, что получилось хорошо, а что плохо, выявить причины и продумать, как исправить ошибки в будущем.

Например, ты сделал татуировку и снова встретился с друзьями. Ты рассказываешь, что тату можно было сделать дешевле, обсуждаешь работу мастера и решаешь, пойдешь ли к нему снова.

DEMO, ИЛИ ДЕМО

Встреча в конце спринта, на которой разработчики показывают процесс и результаты работы другим членам команды. Нужна, чтобы каждый понимал, что результат появляется не на ровном месте, а еще — чтобы быть в курсе, на каком этапе сейчас разработка.

Например, ты проводишь демо, когда показываешь друзьям готовую татуировку, крутишь рукой туда-сюда и акцентируешь внимание на деталях.

Каждую задачу из бэклога спринта выполняют по техническому заданию (ТЗ).

ТЕХНИЧЕСКОЕ ЗАДАНИЕ, ИЛИ TERMS OF REFERENCE (TOR)

Это документ, в котором прописано, зачем поставили задачу, какой она должна быть, какие материалы брать за основу.

Это как рецепт, в котором четко прописано количество ингредиентов, в каком порядке их добавлять, сколько запекать и при какой температуре, какое блюдо получается в итоге.

Техническое задание нужно для верификации — не путай с валидацией.

ВЕРИФИКАЦИЯ, ИЛИ VERIFICATION

Это процесс оценки, насколько система соответствует техническому заданию.

Например, на фармацевтическом заводе работают специальные проверяющие. Они читают технические задания к таблеткам и выясняют, сколько веществ и в каком соотношении в них содержится. А потом сверяют ТЗ с таблетками, которые произвел завод.

А еще перед созданием продукта важно разобраться, зачем он нужен пользователю, какую свою задачу он решит. Для этого составляют User Story.

USER STORY, ИЛИ ПОЛЬЗОВАТЕЛЬСКАЯ ИСТОРИЯ

Описание того, что получает пользователь от продукта.

Например, разработчики пишут приложение для менеджеров, в котором они могут ставить задачи на день и распределять их между сотрудниками.

Тогда история выглядит так: как менеджер, я хочу организовать работу сотрудников, чтобы не перегрузить их задачами и облегчить себе контроль за прогрессом.

Продукт можно создать с нуля, а можно использовать готовый фреймворк, чтобы меньше ошибаться.

ФРЕЙМВОРК

Готовая среда для создания программ, каркас, в который можно вписать нужные функции. Внутри можно делать что угодно, фреймворк — это только инструмент.

Например, можно собрать кухню с помощью готового конструктора на сайте «Икеи», а можно придумать ее с нуля. «Икея» уже учла, сколько тебе нужно розеток, как должны открываться шкафы, поэтому ты не ошибешься и соберешь функциональную кухню из тех материалов, которые захочешь. А если создаешь кухню с нуля, то можешь не учесть расположение вентиляции и вытяжки, прогадать с высотой полок в шкафах и намучиться так, что придется долго исправлять.

Поэтому айтишники часто работают в готовых фреймворках.

API (APPLICATION PROGRAMMING INTERFACE)

Продукты не работают в отрыве от других продуктов.

Например, когда ты оплачиваешь онлайн-заказ в приложении, открывается форма оплаты Сбербанка. Ты вводишь данные карты, код из СМС и оплачиваешь заказ.

Эта форма называется API.

API — это интерфейс программирования приложений, то, что приложение может сделать по запросу других. API «вшивают» в код продукта, чтобы программы работали в связке.