**Немного истории**

Чтобы понять суть докера, сперва углубимся в историю. Раньше доставка грузов по всему миру осуществлялась в бочках, коробках и прочих ёмкостях. Проблема была в том, что все они были разных размеров и форм. Для разгрузки корабля требовалась уйма времени, а время увеличивало стоимость доставки.



Никто не мог ускорить процесс, ведь для разгрузки требовались люди и добавляя больше людей увеличивалась стоимость, добавляя их меньше – увеличивалось время.

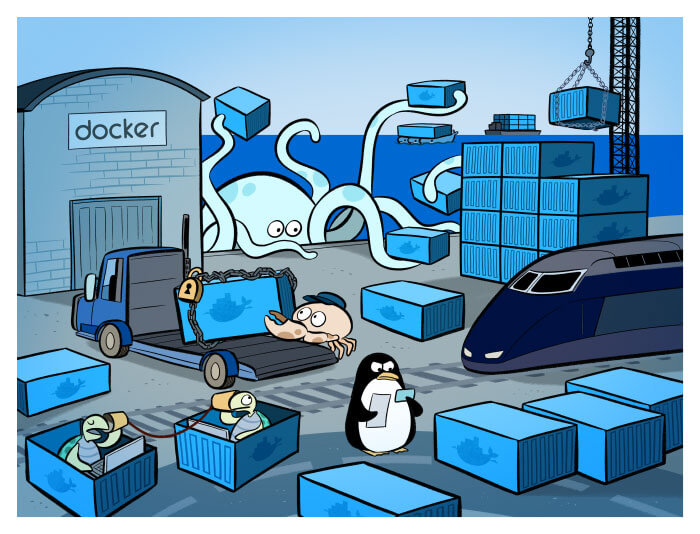
Все было достаточно печально до тех пор, пока не изобрели контейнеры. Те самые контейнеры, что сегодня можно встретить на любом корабле.



Они стали феноменом! Теперь не важно, что конкретно находится внутри самой ёмкости, так как сама ёмкость одной формы. Систематизировав эту, казалось бы, мелочь, люди смогли построить по всему миру одинаковые краны для разгрузки. Тем самым сроки разгрузки и погрузки сократились до минимума.

**При чем здесь Докер?**

Но при чем здесь Докер? Докер сделал точно то же действие, но в программировании. Ранее для открытия проекта требовалось установить много разных технологий и настроить все на устройстве. Вам нужно открыть веб сайт, написанный на PHP? В таком случае установите локальный сервер, настройте Apache, запустите весь проект и о чудо – не получите при этом никаких ошибок. И вроде просто, но что, если вам пришлют какой-то проект с технологией и языком, с которым вы до этого никогда не работали? У вас уйдет много времени чтобы все это протестировать.



Докер кардинально изменил подход, предложив систему контейнеров. Представим, что вы создали проект и сам проект может быть написан на чем-угодно: Питон, Джава, С++ или любой другой язык. Вы запихиваете ваш проект в контейнер, а вместе к проекту добавляете всё необходимое окружение: среду выполнения, сервер, различные пакеты, приложения и всё остальное. Далее при помощи специальных команд вы можете выгрузить ваш проект (контейнер) и после передать его клиенту, начальнику или другому разработчику.

Если вы писали проект на Питон, а отправили его Джава разработчику, то ему не составит труда запустить среду. Процесс запуска всех проектов один и тот же, вне зависимости от того, что находится внутри. Вы можете создать проект, поместить туда что-угодно, а для запуска всегда будет использоваться одна и та же команда, при чем не важно на каком ПК вы будете запускать проект.

**Где применяется Докер?**

Докер применяется повсеместно. Если вы делаете проект для себя, на работе или делаете проект для заказчика – вы повсюду можете использовать Докер.



Если вы отправляете проект без докера, то вам еще потребуется описать процесс настройки проекта и процесс его запуска. При этом никто не исключает, что клиент может ошибиться и у него будет ошибка в ходе тестирования. В таком случае нужно будет уделить время и все показать самостоятельно. Если же воспользоваться Докером, то процесс тестирования свидеться к нескольким вещам: человек должен установить у себя программу Докер и далее через терминал при помощи одной команды сможет запустить ваш проект. При этом нет опаски что что-то пойдет не так. Вся среда будет в самом контейнере, поэтому ничего не потребуется устанавливать или настраивать.

**Приятный бонус**

Еще одним приятным бонусом при использовании Докер может стать тестирование новых технологий. Порой при тестировании нового языка требуется установить много новых программ. Не всегда охота это делать и теперь при помощи Докер это делать не потребуется.

Если вы захотите протестировать у себя на ПК работу WordPress, то не нужно будет устанавливать локальный сервер, на него устанавливать WordPress и далее все настраивать. Вы сможете просто взять  готовый контейнер с установленным WordPress и запустить его. На установку у вас уйдет не более 1 минуты, после чего вы сможете поработать с новой технологией и далее либо полноценно её установить на ПК, либо забыть про нее.

**План курса**

В ходе курса мы с вами научимся работать с контейнерами Докер. Вы научитесь правильно запускать контейнеры у себя на компьютере, научитесь создавать и помещать в контейнер нужную информацию и дополнительно вы изучите все основные команды Докер и познакомитесь с Docker Compose.