SO WHAT IS DOCKER?
CONTAINERIZATION

A Docker container is a portable store for a single component and its dependencies.
SIMILAR TO A LIGHTWEIGHT VM

- Both provide *isolated environments*
- Docker is much more efficient
- 64-bit Linux only (currently)
DOCKER IMAGES

- Like a stopped VM
- Built from Dockerfiles
- List of repeatable steps
- Mongo Dockerfile
BUT DOCKER IS MORE THAN THIS

• Aims to be an "open platform"
• Key to this is the Docker Hub
USE CASES

- Continuous Delivery
  - Reduces difference between dev and production
- Microservice Architectures
  - Rapid Provisioning
  - Rapid Deployment
  - Devops
- Wrapping Applications
THE MOMENT OF EPIPHANY!
SO? WHAT’S THE BIG DEAL?
IT’S JUST A CAT WITH A
POP-TART BODY FLYING
THROUGH SPACE WITH
RAINBOWS COMING OUT
ITS REAR END!
• Easy Install
• (Relatively) Safe
• Portable
• Easy to Throw Away
EASY INSTALL
BUT IT'S IN THE PACKAGE MANAGER!

- But what if it wasn't?
- Or it's not in my package manager?
- Build from source?
  - Runtime dependencies
  - Build dependencies
RELATIVELY SAFE
WHEN WAS THE LAST TIME YOU BLINDLY TRUSTED SOFTWARE?

- Everyday we download and run random software
- Could potentially steal data
- Delete things
- Accidentally or intentionally
SUBUSER

- http://subuser.org
- Secure package manager
- Uses Docker to isolate software packages
- Config file for permissions
  - Access to x11
  - Access to devices
  - Access to folders
- Installs for common software
PORTABLE
- Should run identically everywhere
- Doesn't need a package manager
- Won't be different on Ubuntu and Fedora
- Less testing with different versions of libs
- Can run on a remote machine
  - Docker Hub
EASY TO THROW AWAY
• Uninstallation is a snap
• Dependencies gone too
• No files left lying about
USE CASES
public static void main(String[] args) {
    SchemaFactory factory = SchemaFactory.newInstance("http://www.w3.org/2001/XMLSchema");

    File XSDFile = new File(mXSDFileName);
    File XMLFile = new File(mXMLFileName);

    try {
        Schema schema = factory.newSchema(XSDFile);
        Validator validator = schema.newValidator();

        Source source = new StreamSource(XMLFile);

        try {
            validator.validate(source);
            System.out.println(mXMLFileName + " validates.");
        } catch (SAXException ex) {
            System.out.println(mXMLFileName + " fails to validate because: 
" + ex.getMessage());
            System.out.println();
        } catch (IOException io) {
            System.err.println("Error reading XML source: " + mXMLFileName);
            System.err.println(io.getMessage());
            System.exit(1);
        } catch (SAXException sch) {
            System.err.println("Error reading XML Schema: " + mXSDFileName);
            System.exit(1);
        }
    }
}

/**
 * Checks and interprets the command line arguments.
 */
DEVELOPMENT ENVIRONMENTS

- Share pre-configured set-up
- Plug-ins, source repositories, coding standards
- Also Vagrant
SPECIALIZED SOFTWARE & CONFIGURATIONS

- Sound/Video Editing
- Mail Servers
- Industry/Device Specific
- Stacks
LEGACY SOFTWARE

- Bit rot
- Pin everything to specific version
- Still difficult
  - Docker moving target
  - Kernel
  - Package Repositories
ERR, THIS?

- Fit of rage at Powerpoint
- Decided to try reveal.js
- Hard to install
  - Node, NPM, Grunt...
SIMPLE TO RUN

docker run -d -p 8000:8000 -v /home/adrian/presentations/zen_thru_docker:/revealjs
amouat/revealjs
LINKS

- https://registry.hub.docker.com/u/amouat/revealjs/
- RevealJS Dockerfile
WHY IS THIS COOL?

- Runs anywhere Docker runs
- Can easily host on web server
- Or easily distribute via Hub
SO WHAT WAS THAT ABOUT ZEN?

- Currently writing a book on Docker
- Wanted a "distraction free" writing environment
  - Think WriteRoom, Ommwriter
  - Zen mode for Github
- Decided to use Vim
MAKING DIM

- Had to analyse *dozens* of plug-ins and articles
- Try out various things
- Without breaking existing Vim
- Docker helped!
Containers are an old concept. For decades, UNIX systems have had the chroot command which provides a simple form of filesystem isolation. FreeBSD has had the jail utility since 1998, which extended chroot sandboxing to processes. Solaris Zones, LXC and OpenVZ continued to build on the concept, coming closer to creating completely isolated and self-sufficient environments similar to those provided by VMs. However, it wasn’t until Docker appeared on the scene in 2013 that containers became mainstream.

Docker took the existing Linux container technology added various missing pieces to create a complete and user-friendly solution for creating and distributing containers. The Docker platform has two distinct components; the Docker Engine, which is responsible for creating and running containers and the Docker Hub, a cloud service for distributing containers.

//Need to bring in concept of docker containers and docker infra here

//Perhaps the most significant of these pieces is the use of Union
//file systems, Docker registries and the Docker API.

//Not sure about this bit
//Union file systems enable the layering and caching model used in Docker. This
//model is central to the fast start-up time and dependency sharing provided by
DIM ADVANTAGES

- Quick install
- Doesn't interfere with existing Vim
- Nice to have separate program
DOCKERIZE ALL APPS!

- Easy to distribute & install
- Controlled dependencies
- Safer for users
- Portable
• Chief Scientist @ Container Solutions
• Writing O'Reilly book on Docker
• http://www.container-solutions.com
• @adrianmouat
• adrian@adrianmouat.com