DVS Technologies

DVS Azure DevOps Course Content

INTRODUCTION TO DEVOPS

- What is DevOps?
- History of DevOps
- Different Teams Involved
- DevOps definitions
- DevOps and Software Development Life Cycle
    - Waterfall Model
    - Agile Model
- DevOps main objectives
- Prerequisites for DevOps
- Continuous Testing and Integration
- Continuous Release and Deployment
- Continuous Application Monitoring
- Configuration Management
- What is Cloud?
- History and evolution of cloud
- Cloud Computing Concepts
- Public, Private, Hybrid Clouds
- IAAS, SAAS, PAAS Cloud Models
- Public Clouds
    - Amazon Web Services, Azure, Oracle Cloud, IBM Cloud
- DevOps with Cloud

MODULE 1: OPERATING SYSTEM

LINUX ADMINISTRATION

OVERVIEW

This course teaches the advanced concepts of processes, programs and the components of the Linux operating system. You learn the advanced knowledge of computer hardware, gain an understanding of open source applications in the workplace, and learn to navigate systems on a Linux desktop rudimentary commands to navigate the Linux command line.

INTRODUCTION

- A Linux Introduction
- Open Source Philosophy
- Distributions
- Basic Kernel & Shell Architecture

How And Where To Install Linux

- Using the Linux Essentials Lab Server
- How to Install CentOS 7 with Virtual Box & VMware

Accessing Server

- Usage of putty
- Password less login
- Booting Process

Command Line Basics

- Basic Shell
- Command Line Syntax - Basic Commands
- Ls, pwd, history, cat
- Cp, mv, tail, head
- Touch, cd, find, less
- view, rm, man, chmod
- chown, chgrp, grep, cd,
- cd.., man

Using Directories and Listing Files

- The Linux File System
- Files, Directories
- Hidden Files and Directories

Creating, Moving and Deleting Files

- Files and Directories
- Case Sensitivity
- Simple Globbing and Quoting

Text Editors

- Vi, Vim,nano

Creating Users and Groups and Ownership

- User IDs
- User Commands
- Group Commands
- File/Directory Permissions and Owners

Package and Repository Management

- Package Management Tools and Repositories
- Hands-On - Package Management (YUM, rpm) - For Use with Linux Essentials Lab Servers

Network Management

- Network Configuration
- ethtool,
- ifconfig,
- netstat,
- nslookup,
- ping,
- scp,

File system Management

- Standard Partition
- Logical Volume Management

System Monitoring

- 1.Top 2. Vmstat 3.Sar 4.Ping 5.Ps

Services Management

- Apache (httpd)

Module 2: Linux system level automation

Unix & Linux Shell Scripting

Course Details

Overview:

Shell scripts can take input from a user or file and output them to the screen. Whenever you find yourself doing the same task over and over again you should use shell scripting, i.e., repetitive task automation. Creating your own power tools/utilities.

Introduction and Getting Started

- Course Introduction
- Configuring setup for the shell

Shell Scripting:

- Special Files
- Variables
    - Predifined
    - User defined
    - Null Variables
    - unsetting a variable
    - Storing File Names
    - Storing File contents to a variable
    - Storing Commands to a variable

    - Exporting Variables
    - Permanently assigning the variables
    - Reading A variable
- Basics
    - (shebang) & Importance
    - Comments
    - Executing a script
- Expressions
    - Athematic Operators
    - Logical Operations
    - Relational operators
- Decision Making
    - if
    - if-then-else
    - elif-ladder
- Command Controlled Loops
    - while loop
    - case
    - until loop
    - for loop
    - select loop

- Loop Direction
    - Input Redirection
    - output Redirection
- Other Controlled Loops
    - break
    - continue
    - sleep
- Special Parameters & Values
    - Arguments & Positional Parameters
    - Argument Validation
- Debugging scripts
- Functions

Module 3: Infrastructure as a Code (IaC)

Terraform

Introduction to Orchestration Tools

1. Configuration Management vs Orchestration
2. Mutable Infrastructure vs Immutable Infrastructure
3. Procedural vs Declarative
4. Client/Server Architecture vs Client-Only Architecture
5. Infrastructure As A Code with AWS Using Terraform
6. Installation
7. Creating connection between CLI & Azure API

Working With Terraform

1. HashiCorp Configuration Language
2. Terraform Console and Output
3. Input Variables
4. Breaking Out Our Variables and Outputs
5. Maps and Lookups
6. Terraform Workspaces
7. Null Resources and Local-exec

TERRAFORM MODULES

1. Introduction to Modules
2. Realtime examples including diff azure services

MODULE 4: CONFIGURATION MANAGEMENT

ANSIBLE

OVERVIEW

Ansible continues to gain traction as a powerful, enterprise level configuration and deployment management tool. With it's standardized Playbook formatting and reliance on Python standards, it is easy to use, quick to learn and puts the power of automation at everyone's fingertips. In this course, we will cover Ansible configuration, modules, command line usage and Playbook building. By the time you are done, you will be able to use Ansible to automate and manage your DevOps infrastructure.

INTRODUCTION

- Ansible what is it good for !
- How Ansible works
- What's so great about ansible?
    - Easy-to-Read syntax
- Nothing to install on the remote hosts
- Push-based
- Ansible Scales down
- Built-in modules
- Installing and configuring Ansible
- Setting up a server for testing

YAML BASICS

- Comments
- Variables
- booleans
- lists
- dictionaries
- multilines
- Dict of Dicts
- Short ways to write YAML code

Inventory - Describing your servers

- The Inventory File
- Hosts and ansible.cfg configuration
- Real-time lab environment setup for class
- Password less login configuration
- Other Inventory parameters
    - ansible_ssh_host, ansible_ssh_port,ansible_ssh_user,
    - ansible_connection,ansible_ssh_pass,ansible_shell_type
- Organizing Prod, Syst &Test environments under groups

Ansible Modules

- Command
- File
- Copy
- Template
- Script
- User
- Yum
- Apt
- Service
- Debug

Variables and Facts

- Defining variables in playbooks
- Viewing the values of variables
- Registering variables
- Facts
    - Viewing all facts associated with a server
    - Viewing a subset of facts
    - Local facts
    - Using set_fact to define a new variable
    - Built-in variables
    - Host_vars
    - Inventory_hostname
    - Groups
- Setting variables on the command line
- Precedence

Conditions and loops

- Condition
    - When

- Loops:

- with_items
- with_fileglob
- with_lines
- with_dict
- with_sequence

SPECIAL CASES

- ignore_errors
- local_action
- delegate_to
- serial
- failed_when
- ansible vault

DEBUGGING ANSIBLE PLAY BOOKS

- Limiting Which Hosts To Run
- Debugging SSH Issues
- Debug Mode
- Checking Your Playbook Before Execution
  - Syntax Check
  - List Hosts
  - Listing All the Tasks
  - Check Mode
  - Step
  - Start-at-Task
  - Tagging

SCALE UP YOUR PLAYBOOKS USING ROLES

- Basic Structure of a Role
- Using Roles in our playbooks
- Ansible Galaxy

REAL TIME PROJECTS USING ANSIBLE

- Automating installation and configuration of Apache in test environment
- Automating E2E java application deployment
- Integrating Ansible with Aws for the infrastructure automation

MODULE 5: INTRODUCTION TO AGILE

- Agile key words
- User stories
- Backlogs
- Epic

- Sprints
- Program increment (PI)

MODULE 6: WORKING WITH AGILE IN AZURE

- Introduction to JIRA
- Working with azure boards
- Creating a userstory
- Azure devops process
- Using sprints
- Integrate boards with slack
- Azure devops AD integration
- Azure permissions
- Working with queries
- Working with charts

MODULE 7: REPOSITORY MANAGEMENT WITH AZURE REPOS

GIT & GIT HUB

Overview

Over the length of this course, we will start at the very beginning of revision and source control the way that it is intended to be done using Git client and server. Once we have a firm handle on how to manage our files at the command prompt and in our own repositories, we will take a look at several of the more commercial or public Git hosting sites - Github and Bitbucket. Finally, we will install the Github clone called Gitlab and take a deep dive in how source control can be used in an online environment that supports team collaboration and build automation using Jenkins.

Introduction

- Introduction to Git and Git hub

Installation and Configuration

- Installing Git
- Basic Configuration

Git Basics

- Empty Repositories
- Git Basics
- Git Ignore
- Cloning
- Cloning: Local Repositories
- Cloning: Remote Repositories

Tagging, Branching and Merging

- Tags
- Branches
- Merging

Logging and Repository Auditing

- Git Log

Working with Github

- Introduction to Github
- Secure Communication
- Working with Github

AZURE REPOS

- Git branching strategy
- Working with azure repos
- Pushing branches to azure repo
- Working with pull request
- Working with visual studio & github
- Create branch & push to github
- Visual studio with azure repo
- Integrating azure board with github

MODULE 8: CONTAINERIZATION

Dockers

Course Details

Overview:

This course will explore Docker from the very basics of installation and function to an in depth review of the use cases and advanced features. We will talk about how Docker is architected in order to provide a better understanding of how to manage Linux Containers using the Docker Client. Once we have a good understanding of the basics, we will dive into the advanced use cases and really uncover the power of the entire system. Now Updated for Docker 1.10+ in 2016! GOALS * Introduce and Understand Linux Containers and Application Virtualization * Relate the Architecture of Containers to the Management of Our Applications * Install and Configure Docker for Our Distribution * Explore the Most Common Use Cases for Docker * Understand the Power and Flexibility Application Virtualization Offers

Dockers & Containers

Learning the Basics of Docker

- Introduction to Docker
- Containers vs. Virtual Machines
- Docker Architecture
- The Docker Hub
- Docker Installation
- Creating our First Image
- Working with Multiple Images
- Packaging a Customized Container
- Running Container Commands with Docker
- Exposing our Container with Port Redirects

The Dockerfile, Builds and Network Configuration

- Dockerfile Directives: USER and RUN
- Dockerfile Directives: RUN Order of Execution
- Dockerfile Directives: ENV
- Dockerfile Directives: CMD vs. RUN
- Dockerfile Directives: ENTRYPOINT
- Dockerfile Directives: EXPOSE
- Container Volume Management
- Docker Network: List and Inspect
- Docker Network: Create and Remove
- Docker Network: Assign to Containers

Storage Management

- Storage Overview
- Categories of data storage
    - Non-persistent
    - Persistent
- Persistent Data Using Volumes
- Volume Commands

Monitoring Docker containers & housekeeping

- Monitoring
    - docker stats
    - docker top
- Housekeeping docker containers

Docker Commands and Structures

- Inspect Container Processes
- Previous Container Management
- Controlling Port Exposure on Containers
- Naming Our Containers
- Docker Events
- Managing and Removing Base Images
- Saving and Loading Docker Images
- Image History

Working with docker registry

- Customizing the docker image
- Re-tagging the images
- Pushing customized images to registry

Working with azure container registry:

- Introduction about registry
- Working with az registry
- Generating tokens
- Customize the image & push the image to ACR
- Pull & push images to ACR

MODULE 9: Microservices with Kubernetes (AKS)

Introduction to Kubernetes

- What is Kubernetes
- Design Overview
- Components
- Building Block of Kubernetes:
- Summar

Understanding Kubernetes Architecture

- Kubernetes Cluster Architecture
- Kubernetes API Primitives
- Kubernetes Services and Network Primitives
- Workernodes
- Masternodes
- PODS
- labels
- Selectors
- Controllers
- Services

Introduction to YAML

- What is YAML
- Structure

Installation, Configuration, and Validation

- Building the Kubernetes Cluster
- Installing Kubernetes Master and Nodes
- Building a Highly Available Kubernetes Cluster
- Running End-to-End Tests on Your Cluster

Cluster Communications

- Pod and Node Networking
- Container Network Interface (CNI)
- Service Networking

Configuring the Kubernetes Scheduler

- Labeling the nodes
- Node affinity
- NodeSelector
- Resource Management
- DaemonSets and Manually Scheduled Pods
- Displaying Scheduler Events

Application Lifecycle Management

- Deploying an Application,
- Rolling Updates
- Rollbacks

Configuring an Application for High Availability and Scale

- Creating a Configmap
- Mounting Configmap as a volume
- Creating a secret
- Mounting secret as a volume

Storage Management

- Persistent Volumes
- Volume Reclaim Policies
- Volume Access Modes
- Create Persistent volume
- Create Persistent volume Claim

Logging and Monitoring

- Monitoring the Cluster Components
- Monitoring the Applications Running within a Cluster
- Liveness Probe
- readiness Probe
- Managing Application Logs
- Getting pod logs
- Auto Scaling

AZURE KUBERNETES SERVICE

- Deploying azure kubernetes service
- AKS using CLI
- Cluster login
- Add user node pools
- Taint & tolerations
- Adding ACR to AKS
- Ingress

MODULE 10: CONTINUOUS INTEGRATION – PART 1

- Explain in detail about app environments
- what is ci & build process
- Checking free tier limits
- Working with projects(public, private)
- Configuring our own agents
- Preparing infra for azure CI
- Configuring and testing PR pipeline
- Configuring and testing build pipeline
- Executing build pipeline
- Create and Publish code to azure repo
- Create and import git repo to azure repo
- Configure pipelines with az agents
- Integrate whitesource bolt with azure devops
- Integrate sonarcloud with az devops
- Configure pr pipelines with az devops
- Integrate azure pipelines with az slack
- Working with classic editor
- Pipeline analytics

MODULE 11: CONTINUOUS DELIVERY

- What is Continuous delivery and deployment
- Configuring basic build pipeline
- Build and deploy the .net core application
- Configure build triggers
- Deploying application in multiple environments
- Approvals & Gates
- Implementing gates using azure policies

MODULE 12: CONTINUOUS DELIVER WITH DOCKERS

- Introduction to docker
- Build & deploy .net core in docker
- Working with azure container registry(ACR)

- Working with azure container instance
- Working with azure kubernetes service (AKS)
- Build image using azure build pipeline
- Deploying image to AKS using build pipelines
- Build and deploy using release pipelines on app service
- Az release pipeline – AKS deployment
- Working with helm charts

MODULE 13: AZ PIPELINES AND DEPLOYMENT STRATEGIES

- Working with azure keyvault
- Azure pipeline variable groups
- Configure variable groups with key vault
- Using variables with release pipelines
- Deployment strategies
- Prepare infrastructure
- Configure pipelines
- Azure traffic manager

MODULE 14: AZURE ARTIFACTS

- Introduction about artifacts & azure artifacts
- Creating the feed
- Implementing versioning
- Deploy artifacts with versioning

MODULE 15: FINAL PROJECT: JAVA APPLICATION DEPLOYMENT WITH AZURE DEVOPS

Project Overview:

- In this final project, you will build and deploy a Java-based web application using Azure DevOps. You will implement Continuous Integration (CI) and Continuous Deployment (CD) pipelines to automate the build, test, and deployment processes. Additionally, you will utilize Ansible and Terraform for infrastructure automation, managing code repositories with Git and Azure Repositories.

Project Requirements:

Java Web Application Development:

- Develop a Java-based web application (e.g., Spring Boot
- Ensure the application is container-ready by implementing Docker support

Azure DevOps Setup:

- Create an Azure DevOps organization and project.
- Configure service connections to Azure resources (e.g., Azure Container Registry, Azure Kubernetes Service).
- Set up appropriate permissions and access controls for team members.

CI Pipeline Setup:

- Implement a CI pipeline in Azure DevOps to build the Java application.
- Configure the pipeline to trigger on code commits to the main branch.
- Include steps for compiling the code, running unit tests, and generating artifacts (JAR or WAR files).

PR Pipeline Configuration:

- Implement a Pull Request (PR) pipeline for code reviews.
- Configure the PR pipeline to perform code analysis, including static code analysis and security scans.
- Ensure that the pipeline provides feedback to developers before merging code into the main branch.

CD Pipeline Deployment:

- Set up a CD pipeline for automated deployment of the Java application.
- Integrate the CD pipeline with Docker to build container images for the application.
- Deploy the application to Azure Kubernetes Service (AKS) using the CD pipeline.

Infrastructure Automation with Ansible and Terraform:

- Use Terraform to define the infrastructure components required for deploying the Java application (e.g., AKS cluster, Azure Database for PostgreSQL).
- Configure Ansible playbooks for application deployment and configuration management.
- Integrate Ansible and Terraform with the CD pipeline to automate infrastructure provisioning and application deployment.

Integration Testing and Monitoring:

- Implement integration tests to ensure the deployed application functions correctly.
- Set up monitoring and logging for the application using Azure Monitor and Azure Log Analytics.

Documentation and Presentation:

- Document the project setup, including CI/CD pipeline configurations, infrastructure code, and deployment process.

- Prepare a presentation to showcase the project's architecture, deployment workflow, and key learnings.
- Project Deliverables:
- Deployed Java web application on Azure Kubernetes Service.
- Configured CI/CD pipelines in Azure DevOps for automated build, test, and deployment.
- Infrastructure automation scripts using Ansible and Terraform.
- Documentation detailing project setup, pipeline configurations, and deployment process.
- Presentation slides highlighting the project architecture, workflow, and lessons learned.
- Project Evaluation Criteria:
- Completeness and correctness of CI/CD pipelines.
- Efficiency of infrastructure automation using Ansible and Terraform.
- Deployment success rate and application stability.
- Quality of documentation and presentation.

Conclusion:

- This final project will provide hands-on experience in deploying Java applications using Azure DevOps and modern DevOps practices. By integrating CI/CD pipelines and infrastructure automation, you will streamline the development process and enhance the reliability of deployments.