

CI/CD

UDA PEOPLE

```
object to mirror  
mirror_mod.mirror_object =  
operation == "MIRROR_X":  
mirror_mod.use_x = True  
mirror_mod.use_y = False  
mirror_mod.use_z = False  
operation == "MIRROR_Y":  
mirror_mod.use_x = False  
mirror_mod.use_y = True  
mirror_mod.use_z = False  
operation == "MIRROR_Z":  
mirror_mod.use_x = False  
mirror_mod.use_y = False  
mirror_mod.use_z = True
```

```
selection at the end -add  
mirror_ob.select= 1  
mirror_ob.select=1  
context.scene.objects.active  
("Selected" + str(modifier))  
mirror_ob.select = 0  
 bpy.context.selected_objects  
 objects[one.name].select  
print("please select exactly")
```

```
-- OPERATOR CLASSES --
```

```
types.Operator):  
X mirror to the selected  
object.mirror_mirror_x"  
mirror X"
```



WHAT IS CI/CD STANDS FOR ?

consist of three major concepts

- ❑ **Continuous Integration:** describes the process of merging developer branches to the main branch several times a day. CI puts an emphasis on test automation and finally generates a high quality, deployable artifact.
- ❑ **Continuous Delivery:** In addition to Continuous Integration, Continuous Delivery makes sure that changes of a software product can be released quickly to customers in an automated way and at any point in time.
- ❑ **Continuous Deployment** extends Continuous Delivery in such a way that it allows frequent automated deployments without any human interaction. Typical phases in Continuous Deployment are Infrastructure Provisioning, Smoke Testing, Production Deployments and automated Rollbacks.



WHAT IS CI/CD ?

- ❑ CI/CD is a method to frequently deliver apps to customers by introducing automation into the stages of app development. The main concepts attributed to CI/CD are continuous integration, continuous delivery, and continuous deployment.
- ❑ CI/CD is a solution to the problems integrating new code can cause for development and operations teams
- ❑ Specifically, CI/CD introduces ongoing automation and continuous monitoring throughout the lifecycle of apps, from integration and testing phases to delivery and deployment. Taken together, these connected practices are often referred to as a "CI/CD pipeline"

A series of four parallel white diagonal lines in the bottom-left corner of the slide.

WHAT IS CONTINUOUS INTEGRATION ?

- Compile
- Unit Test
- Static Analysis
- Dependency vulnerability testing
- Store artifact



WHAT IS CONTINUOUS DEPLOYMENT ?

- Creating infrastructure
- Provisioning servers
- Copying files
- Promoting to production
- Smoke Testing
- Rollbacks



THE BENEFITS

- TECHNICAL BENEFIT

- Catch Compile Errors After Merge
- Catch Unit Test Failures
- Deploy to Production Without Manual Checks
- Automate Infrastructure Creation
- Automated Smoke Tests

- BUSINESS BENEFIT

- Less developer time on issues from new developer code
- Less bugs in production and less time in testing
- Less human error, Faster deployments
- Less time to market

A series of four parallel white diagonal lines in the bottom-left corner of the slide.

CONFIGURATION MANAGEMENT TOOL

- INSTALLABLE/ON-PREM

- Catch compile error after merge
- Catch unit test failure
- Automatic infrastructure creation
- Automated Smoke test



CONFIGURATION MANAGEMENT TOOL

- CLOUD-BASED

- Bambo
- Circle CI
- Travirs CI
- Gitlap



MONITORING

- MONITORING TOOLS

- Graphity
- Loggely
- Cloud watch
- prometheus
- Datadog
- logstash