Table of Contents

Anyframe ODEN

version 2.6.0

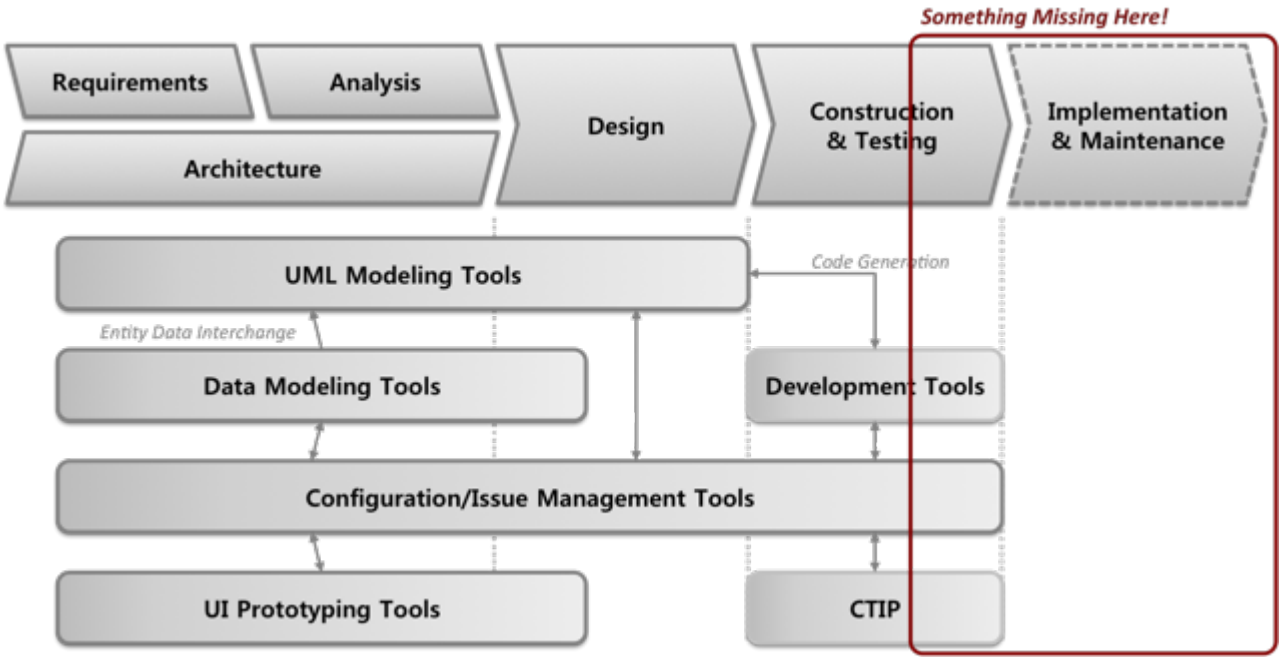
2015.12.02

Chapter 1. Overview

Anyframe Open Deployment ENvironment (이하 ODEN)Anyframe Open Deployment ENvironment (이하 ODEN) [1: ODEN은 OSGi, Eclipse 등 Java 기술을 활용하고 있으며, 손쉽게 확장이 가능한 개방형 아키텍처를 지향한다. 더불어 각종 UNIX, Mac OS X, Microsoft Windows 등 다양한 플랫폼 환경을 지원한다.]은 CI은 CI [2: Continuous Integration: 빌드, 테스트, 배포 등의 일련의 절차를 통합/자동화함으로써 소프트웨어 딜리버리의 시간을 단축시키는 기법 및 이를 가능하게 하는 것]환경을 통해 빌드된 어플리케이션 컴포넌트 및 각종 설정파일, 웹파일 등을 원하는 배포대상서버에 편리하게 배포할 수 있도록 하는 배포관리 툴이다. 본 장에서는 ODEN의 개발 배경 및 주요 특징에 대한 간략한 소개를 제공한다.

1.1. Introduction

최근의 일반적인 중대형 개발 프로젝트의 프로세스는 요구정의로부터 시작하여 분석 및 아키텍처정의, 설계, 개발, 이행 등으로 진행하는 것이 최근의 추세이며, 여러 벤더들은 이러한 각 공정 단계에 특화된 다양한 툴을 제공하고 있다. 그러나 분석/아키텍처정의/설계 단계에서 활용할 수 있는 다양한 툴과는 달리, 개발단계 이후 활용할 수 있는 툴의 범위는 다소 부족한 듯 하다. 특히, 개발한 어플리케이션 컴포넌트들을 개발서버 및 테스트서버 혹은 그 너머의 운영서버 등에 배포할 수 있는 전문화된 배포관리 툴에 대한 선택의 폭은 매우 작은 것이 현실이다.



이러한 전문적인 배포관리툴의 부재로 인해, 현장 프로젝트에서는 몇가지 어려움에 직면하게 된다.

첫째, 자동화 및 정형화된 배포관리가 이루어지지 않아 프로젝트 진행 및 운영 시 추가적인 리소스가 필요하게 된다.

- 기존에는 수작업 또는 CI엔진에 의한 복사 등으로 배포를 실시하였는데, 이를 관리하기 위해 QAO기존에는 수작업 또는 CI엔진에 의한 복사 등으로 배포를 실시하였는데, 이를 관리하기 위해 QAO [3: Quality Assurance Officer: 품질관리자]혹은 SA혹은 SA [4: Software Architect: 소프트웨어 아키텍트]에 의한 배포관리가 전문적으로 이루어져야 했음

- Target Server가 여러대일 경우, 해당 작업을 단순반복하여 처리해야 하므로 번거로운 작업을 수행해야 함

둘째, 배포 시 고려할 수 있는 다양한 배포 방법에 일일이 대응하기가 어렵게 된다.

- 전문적인 배포관리 툴이 없다면 전체 배포, 원하는 것만 배포, 변경된 사항만 배포 등 현장에서 요구하는 다양한 배포 방식에 일일이 대응하기 어려움
- 특히, 개발서버에서는 대개 변경된 사항만 배포되면 족함에도 불구하고, 일일이 비교하는 것이 번거롭기 때문에 전체를 한꺼번에 배포할 경우가 많음
- 실제로 일일이 비교하여 배포하는 경우라도, 누락되는 것이 있어 결국 배포에 실패하게 되는 경우가 발생함

셋째, 표준화 및 정형화된 프로세스에 기반한 개발 및 이행단계 진행이 어렵게 된다.

- 프로세스화된 배포 환경의 부재는 매번 배포 시마다 업무의 혼란 및 리소스의 낭비 여지를 제공하게 됨
- 이를 위해 배포 정책 설정, 스냅샷/롤백, 로그분석, 스케쥴링/배치, 워크플로우 적용 등 다양한 기능들이 필요함

ODEN은 이러한 어려움을 극복하기 위한 자동화된 배포관리 환경을 제공한다.

1.2. Key Features

ODEN은 다음과 같은 주요 특징을 지닌다.

첫째, 현장 프로젝트의 다양한 상황에 대응하기 위한 개방적이고 유연한 구조를 지향한다.

- 현장의 추가기능 요구에 유연하게 대응할 수 있도록 플러그인 형태(OSGi Bundle)로 기능을 추가할 수 있는 플랫폼을 활용하였음
- Http프로토콜을 통해 명령어 형태로 ODEN과 연계할 수 있어, 기존 이관 시스템과의 연계가 편리

둘째, 다양한 형태의 배포 방법을 지원한다.

- 일괄 배포
- 변경된 것만 배포

셋째, 배포관련 다양한 부가기능 및 가이드를 제공한다.

- 이력 조회 기능
- 서버간 정합성 검증 기능

넷째, 다양한 배포환경에 적용할 수 있도록 안정적인 성능을 보장한다.

- 대량, 대용량 배포물에 대한 안정적인 배포실행 제공
- 물리적 디스크 I/O를 최대한 줄여 속도 확보

다섯째, 다양한 인터페이스를 제공한다.

- CLI 환경을 통한 배포 실행 및 모니터링

- Web UI를 통한 배포 실행 및 모니터링

여섯째, CI서버와 연동을 통해 빌드수행이 가능하다.

- Jenkins와 연동 가능

1.3. Support

1.3.1. 기술 지원

ODEN에 대한 기술 지원은 Anyframe 오픈소스 커뮤니티 사이트의 포럼 (http://www.anyframejava.org/forum)을 통해 이루어지며, 단순 질의 응답에서부터 소스 코드에 대한 구체적인 가이드 및 해결책을 제시한다. 특정 프로젝트를 위한 기술 컨설팅 지원이 필요한 경우 Anyframe 오픈소스 커뮤니티 사이트의 연락처(http://www.anyframejava.org/about/contactus)를 통해 요청할 수 있다. 또한, 이슈관리시스템인 JIRA(http://dev.anyframejava.org/jira)를 통해, Bug Fix, Improvements, New Features에 대한 이슈들을 요청할 수 있다. 자세한 사용방법은 이곳(http://www.anyframejava.org/development/issue)을 참조하도록 한다.

1.3.2. 유지 보수

Anyframe 오픈소스 커뮤니티 사이트(http://www.anyframejava.org)를 통해 릴리즈된 최신버전 및 이전 버전에 해당하는 라이브러리, 매뉴얼 등을 제공받을 수 있으며 패치 및 업그레이드되는 파일의 경우에도 오픈소스 커뮤니티 사이트를 통해 제공된다. 오픈소스 커뮤니티 사이트의 첫페이지를 통해서 공지되므로 필요 시 참조하도록 한다.

Chapter 2. Concepts

본 장에서는 ODEN에 대한 보다 상세한 이해를 돕기 위해, ODEN에서 지향하는 아키텍처의 형태 및 주요 구성요소에 대한 소개를 제공한다.

2.1. Architecture

ODEN을 활용한 배포관리 환경은 크게 다음과 같은 구성요소로 이루어진다.

- Server and Agent

배포관련 각종 작업을 수행하는 핵심모듈로, Server는 빌드서버에, Agent는 Target Server(파일이 배포될 서버)에 설치되며, CLI 기반 UI를 제공함

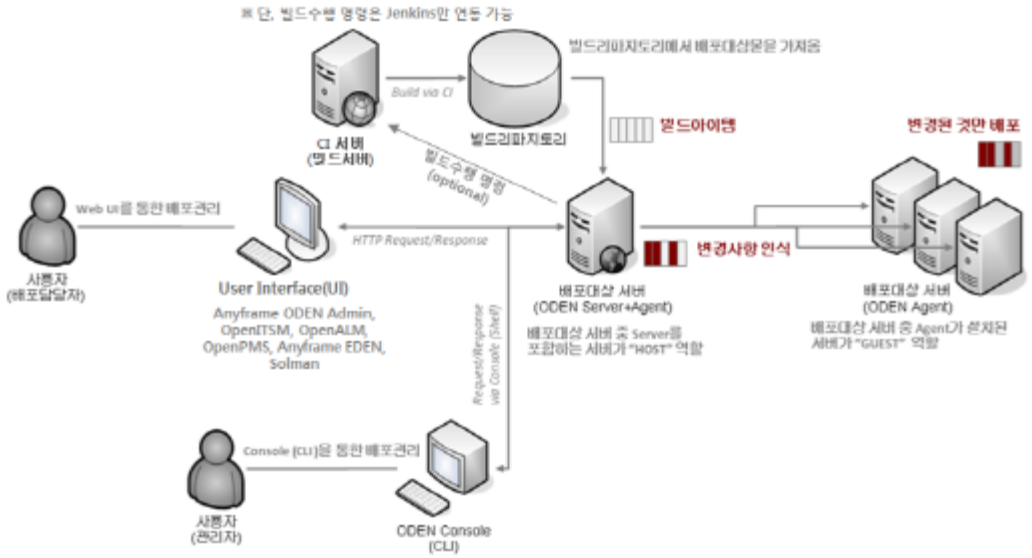
- Oden Admin

배포 모니터링 및 실행을 위한 GUI 환경

ODEN을 활용하여 배포대상물을 원하는 Target Server에 배포하고, 로그를 통해 결과를 조회할 수 있다. 이러한 일련의 내용을 각 구간별로 설명하면 다음과 같다.

- CI서버를 통해 배포대상물을 빌드(Jekins와 연동 가능)
- 명령어 혹은 ODEN Admin을 통해 배포 요청
- 배포대상물에 대한 변경감지하여 변경대상을 Agent로 전달
- Agent를 통해 배포 및 배포내역 저장
- 명령어 혹은 ODEN Admin을 통해 배포결과 확인

다음 그림은 이러한 배포 흐름을 개념적으로 설명한 것이다.



2.2. Servers and Agents

ODEN의 Server 및 Agent는 배포작업을 실질적으로 관할하고 수행하는 역할을 수행한다.

Server 및 Agent는 배포할 것들을 확인, 감지, 목록화, 전송, 배포하는 일련의 작업들을 수행하며, 이러한 것들은 각각의 기능구성 및 설치 환경 등을 고려하여 개별 번들Server 및 Agent는 배포할 것들을 확인, 감지, 목록화, 전송, 배포하는 일련의 작업들을 수행하며, 이러한 것들은 각각의 기능구성 및 설치 환경 등을 고려하여 개별 번들 [5: ODEN Server 및 Agent는 OSGi 표준사양을 따르는 개별 번들로 구성되어 있다.]로 구성되어 있다.

2.2.1. Servers

ODEN Server는 각종 UI 환경과의 연계를 위한 서비스를 제공한다. 사용자는 ODEN UI 를 통해 ODEN Server로 명령을 내리고, ODEN Server는 Agent에 적절한 요청을 보낸다.

ODEN Server는 배포할 파일이 있는 시스템에 설치하여야 한다. 시스템에서 배포대상 파일을 읽어 Agent로 전송한다.

2.2.2. Agents

ODEN Agent는 ODEN Server로부터 전달받은 배포대상물을 지정된 위치에 배포 및 Server로 부터 받은 각종 요청을 수행한다.

파일이 배포될 서버에 각각 한 개씩 설치해야 한다.

2.3. Deploying Items

ODEN은 배포할 파일이 존재하는 위치(Source)와 파일이 배포될 경로들(Target)을 묶어 Job형태로 관리한다. 배포 실행은 Job단위로 이루어 진다.

2.3.1. Job

어디에서 어디로 배포할 것인지에 대해 기술된 배포 설정. 하나의 Source와 여러개의 Target으로 이루어 짐.

2.3.2. Source

배포할 파일이 존재하는 경로. ODEN Server가 설치된 시스템에 존재.

2.3.3. Target

파일이 배포될 위치를 말함. 해당 시스템에 Agent가 설치되어 있어야 함.

2.4. User Interface

ODEN은 기본적으로 CLI(Command Line Interface) 환경을 제공하며, 사용의 편리를 위해 별도의 GUI환경을 제공한다.

2.4.1. Command Line Interface

ODEN Server에서 제공하는 스크립트를 통해 ODEN으로 명령어를 전달할 수 있다. 보다 자세한 사항은 Working with Command Line을 참고한다.

2.4.2. Graphical User Interface

ODEN의 보다 용이한 사용을 위해 ODEN Admin이라는 GUI환경을 제공한다.

Chapter 3. Install and Configuration

본 장에서는 ODEN을 사용하기 위한 설치요구사항과 설치 및 설정에 대한 소개를 제공한다.

3.1. System Requirements

ODEN을 설치하기 위해서는 다음의 요구사항을 충족해야 한다.

- ODEN Server 및 Agent 설치요구사항
  - Java Runtime Environment 1.5 or above
  - 20MB 이상 하드디스크 공간
  - 512MB RAM 이상
- ODEN Admin 설치요구사항
  - Java Runtime Environment 1.5 or above
  - 30MB 이상 하드디스크 공간
  - 256MB RAM 이상
- Linux, UNIX, Solaris, Mac OS X, MS Windows 등의 플랫폼
- 서버간 방화벽 확인 (default Port 변경가능)
  - ODEN Server(Port:9860) -> ODEN Agent(Port:9872)
  - 사용자PC -> ODEN Admin(Port:9880)
  - ODEN Sever(Port:9860) -> Jenkins(Port:9090)

3.2. Installing The ODEN

3.2.1. Server and Agents

ODEN은 Server모드와 Agent모드가 존재한다. Agent는 파일이 배포될 서버인 Target ServerODEN은 Server모드와 Agent모드가 존재한다. Agent는 파일이 배포될 서버인 Target Server [6: 이 문서에서는 파일이 배포될 서버를 Target Server라 칭한다.]에 설치되어 ODEN Server에서 지시하는 명령에 따라 파일을 배포한다. ODEN Server는 여러개의 Agent를 관리하며 사용자는 커맨드 라인이나 ODEN Admin과 같은 GUI환경을 통해 ODEN Server에 명령을 내리게 된다. 사용자는 Agent에 직접 명령을 내릴 수 없으며, ODEN Server를 통해 Agent에 명령을 내려야 하며 대칭키 방식의 알고리즘을 통해 암호화 하여 주요 정보는 기밀성을 보장한다.

3.2.1.1. Installing Server and Agents

배포될 파일이 존재하는 시스템에 ODEN Server를 설치해야 하고 원활한 작동을 위해 설치 경로는 영문으로 구성되어야 한다.

ODEN의 압축을 풀면 아래와 같은 구조로 되어 있다.

```
+---- core
|    |
|    +---- bin        // 실행 스크립트
|    |
|    +---- bundle     // Oden 및 라이브러리
|    |
|    +---- conf       // Oden 설정 파일
|
+---- admin           // Oden Admin 파일 및 실행 스크립트
|
+---- manual          // Oden, Oden Admin 매뉴얼
```

압축파일에는 ODEN 및 ODEN Admin이 모두 포함되어 있다. ODEN은 Server모드(bin/startup.sh or bin/startup.cmd)로도 Agent모드(bin/startup-agent.sh or bin/startup-agent.cmd)로도 구동시킬 수 있다. 구동된 Server/Agent 중지를 원할 경우는 bin/ *-shutdown.sh or bin/ *-shutdown.cmd 를 실행한다. Agent만 설치하고 싶다면, core 디렉토리만 복사하여 Agent모드로 부팅시키면 된다.

ODEN Admin은 ODEN Server와 같은 시스템에 설치되어야 한다. bin/startup.sh(bin/startup.cmd)로 바로 실행시킬 수도 있고, WAS의 application형태로 사용할 수도 있다.

ODEN Agent로 동작시에는 conf/oden.ini파일 대신 conf/agent.ini파일을 참조하여 동작한다.

3.2.1.2. Configuring Server and Agents

conf/oden.ini를 통해 ODEN Server의 설정을 변경할 수 있다. 변경 내용을 적용하기 위해서는 ODEN을 재구동 시켜야 한다. (Agent의 경우 conf/agent.ini 를 통해 일부 옵션을 지원한다.)

- bundle.libs: ODEN library 목록. bundle 폴더에 있는 jar 파일중 이 목록 외의 파일만 active 상태가 되며, 이 목록에 설정된 파일은 비활성화 상태가 되어 library 로써만 사용된다. 일반적으로 사용자가 수정할 필요는 없다.
- cmd.available: 사용가능한 명령어 목록. 수정할 필요 없음.
- log.level: log파일에 뿌려질 로그 레벨(1 = error, 2 = warning, 3 = info, 4 = debug). 로그파일은 meta폴더에서 확인할 수 있다. ODEN에 문제가 생겼을 시 디버깅하기 위한 용도로 사용되며, 일반적으로 수정할 필요는 없다.
- http.port: ODEN과 통신할 때 사용되는 포트.
- deploy.undo: undo 기능 사용 여부. deploy.tmpfile도 true로 설정되어야함.
- deploy.tmpfile: 배포 시 임시파일을 사용할 지 여부. true로 설정되면 배포 시 임시파일을 만들어 전송한 뒤, 기존파일을 대체함. 속도가 느림.
- deploy.undo.loc: 배포전 기존 파일 백업 위치.

- deploy.backupcnt: 배포전 기존 파일 백업 디렉토리 최대 개수.
- deploy.threadcnt: 일괄 배포 시 구동되는 thread 개수. 설정하지 않을 경우, JVM 상에서 유효한 processors를 기준으로 ODEN 에서 자동 설정
- deploy.exception.option: 배포 오류 발생 시 원복 여부. true로 설정되면 배포 오류 발생 시 배포 작업이 취소. false로 설정되면 오류 파일은 제외하고 나머지 파일 배포작업 수행.
- console.user: 커맨드라인으로 명령어 실행 시 사용할 user id. conf/accounts.txt에 미리 등록되어 있어야 함.
- deploy.readtimeout: Server에서 Agent로 요청시 응답이 오기까지 대기하는 시간. 기본 120초. deploy.undo가 활성화되어 있을 경우, 백업으로 인해 응답이 오기까지 시간이 오래 걸릴 수 있다.
- page.scale: page처리 시 사용될 기본값. 수정할 필요없음.
- exec.timeout: 커맨드 명령어 실행 시 대기할 최대 시간. 단위 ms. 설정하지 않을 경우, 커맨드를 강제 종료시키지 않고 끝나길 기다린다.
- log.duration: 배포 이력 보관 기간. 단위 day. 기본값 365.
- server.ip: ODEN Server의 IP. 허용되는 ODEN Server의 IP를 등록할 때 사용
- security.key: 파일 전송 시 암호화 및 복호화를 하기 위한 키 값. 기능을 사용 할 경우 키 값을 정의하며, oden.ini 와 agent.ini 에 동일한 값이 정의 되어야 한다.
- boot.only: 배포 대상 서버에 물리적으로 동일한 공간에서 다수의 Application이 동작 할 경우 하나의 Agent 기동을 원할 경우의 설정값. agent.ini 에 정의하며, Window Server의 경우 Subinacl 윈도우 유틸리티가 설치 되어야 한다. 또한 모든 경로의 접근을 위해 Admin 권한으로 Agent를 구동하여야 한다.
- build.url: Jenkins와 연동을 위한 Jenkins접속 url. ex) http://127.0.0.1:9090

Jenkins와의 연동을 위해서는 추가적으로 아래의 작업이 필요하다. Jenkins의 ./jenkins/war/WEB-INF/web.xml에 아래내용을 추가한다.

```
<servlet>
<servlet-name>Ctip Gen Servlet</servlet-name>
<servlet-
class>org.anyframe.ide.ctip.integration.CtipGenServlet</serv
let-class>
<init-param>
<param-name>hudsonHome</param-name>
<param-value>../jenkins</param-value>
</init-param>
<init-param>
<param-name>hudsonJobDir</param-name>
<param-value>../jenkins/jobs</param-value>
</init-param>
</servlet>
<servlet-mapping>
<servlet-name>Ctip Gen Servlet</servlet-name>
<url-pattern>/anyframe/api/*</url-pattern>
</servlet-mapping>
```

anyframe-ide-ctip-integration-2.7.0.jar 다운받아 ./jenkins/war/WEB-INF/lib 경로에 추가한다.

3.2.1.3. Starting Up Server and Agents

ODEN Server 및 Agent의 설치 및 설정이 끝났으면, 아래의 명령으로 ODEN을 Server모드로 구동시킬 수 있다.

```
bin/startup.sh
(Windows 에서는 ODEN core의 bin폴더로 이동한 뒤 startup.cmd를 실행한다.)
```

아래의 명령으로 ODEN을 Agent모드로 구동시킬 수 있다.

```
bin/startup-agent.sh
(Windows 에서는 ODEN core의 bin폴더로 이동한 뒤 startup-agent.cmd를 실행한다.)
```

ODEN Server를 실행시킨 뒤 커맨드라인을 통해 ODEN 명령어를 실행시킬 수 있다.

```
>cd bin
>runc.sh help
(Windows에서는 runc.cmd help)
```

Important

Server 및 Agent 실행 시, Target Server의 파일 및 디렉토리에 대한 소유자(ownership) 및 권한(permission)에 유의한다. ODEN을 통해 배포한 파일은 Agent를 구동시킨 사용자의 소유 및 권한(umask)를 갖게 된다. 배포할 폴더에 Agent를 구동시킨 사용자가 접근 권한이 없다면 배포는 실패하게 될 것이다.

ODEN이 설치되는 경로에 공백이 있으면 안된다. Windows의 바탕화면이나 내폴더의 경우, 중간에 공백이 있는 경로가 있으므로 ODEN이 제대로 동작하지 않을 것이다.

3.2.1.4. Managing Accounts

bin폴더의 acc.sh(acc.cmd) 명령어로 계정 추가할 수 있다. 기본적으로 제공되는 계정은 ID는 oden, PWD는 oden0이다.

```
acc.sh <id> <pwd>
(Windows에서는 acc.cmd <id> <pwd>)
```

conf폴더의 accounts.txt에서 추가된 계정을 확인할 수 있다. 암호는 인코딩 되어 있다. 해당 라인을 삭제하면 계정이 삭제된다.

명령어로 배포 시 어느 계정을 사용할 것인지 지정해 주어야 한다. conf폴더의 oden.ini의 console.user필드에 지정해 주어야 한다. 기본값은 oden이다. oden계정을 삭제하였을 경우, console.user값을 다른 계정으로 꼭 변경해 주어야 한다.

ODEN Admin에서는 별도의 계정관리 기능이 있으며 본장의 계정관리는 웹콘솔을 위한 계정관리이다.

3.2.2. ODEN Admin

3.2.2.1. Installing ODEN Admin

별도의 WAS없이 독립적으로 ODEN Admin을 사용할 수도 있고, 새 WAS의 application으로 ODEN Admin을 사용할 수도 있다.

별도 WAS없이 사용

```
startup.sh
(Windows에서는 startup.cmd)
```

admin폴더의 startup.sh나 startup.cmd를 실행시키면 자동으로 ODEN Admin이 시작된다.

```
http://localhost:9880
```

위 주소로 Admin에 접근 가능하다. 기본 포트는 9880이다. 포트를 변경하려면 스크립트 파일(startup.sh 혹은 startup.cmd)을 열어 --httpPort의 값을 원하는 포트값으로 수정하여 준다.

Admin 내부에서 사용하는 db 및 ODEN Server 포트를 변경하려면 스크립트 파일(startup.sh 혹은 startup.cmd)을 열어 oden.db.port 및 oden.port의 값을 원하는 포트값으로 수정하면 된다.

별도 WAS에 설치

Tomcat을 기준으로 설명한다. (Tomcat 6.0 이상이 anyframe.oden.admin.war를 WAS의 webapps폴더로 복사한다. Tomcat을 부팅시키면 아래의 주소로 ODEN Admin에 접근할 수 있다.

```
http://localhost:8080/anyframe.oden.admin
```

Chapter 4. Working with Command Line

ODEN 명령어를 실행시키기 위해 runc라는 커맨드라인 명령어를 제공한다. runc.sh(runc.cmd) 이후에 ODEN명령어를 입력하면 ODEN에 명령을 전달할 수 있다.

4.1. Introduction to ODEN Command Line

4.1.1. How to run ODEN Commands

bin폴더의 runc.sh(혹은 runc.cmd) 명령어를 통해 ODEN 명령어를 커맨드라인에서 실행시킬 수 있다.

```
bin/runc.sh help
(Windows에서는 bin/runc.cmd help)
```

웹콘솔을 통해 명령어를 실행할 수 있다.

```
http://localhost:9860/wconsole.html
```

9860은 ODEN Server의 기본 포트이다. 웹콘솔은 Oden Admin이 아닌 Oden Server에서 제공하는 것이다.

4.2. ODEN Commands

ODEN 명령어를 통해 ODEN Server에 명령어를 전달할 수 있다. 인자에 스페이스가 들어가 있으면 " "로 묶어줘야 한다. 옵션 뒤에 여러개의 인자를 넣길 원할 경우 스페이스로 구분하여 나열하면 된다. ""로 묶인 부분은 스페이스가 있더라도 하나의 인자로 인식한다.

4.2.1. Job Command

Job은 어디에서 어디로 배포될 것인지가 기술된 배포 설정이다. 배포를 위해서는 Job이라는 배포설정을 미리 만들어 두고 Job을 실행시켜 배포를 하게 된다. Job은 하나의 Source와 여러개의 Target으로 구성되어 있다. Source는 배포할 파일이 위치한 경로(빌드서버)를 지칭하며, Target은 파일이 배포될 경로(개발 혹은 운영서버)를 말한다. Job을 설정하는 명령어는 별도로 제공하지 않으며 ODEN Server의 jobs.xml을 직접 수정하거나 ODEN Admin을 통해 설정해야 한다. jobs.xml을 수정하는 법은 아래 '참고'에 기술되어 있다.

4.2.1.1. job info

등록되어 있는 job을 확인하기 위한 명령어.

```
job info [ <job> ]
```

job이름이 없을 경우 등록되어 있는 job 목록을 출력. job이름을 인자로 주었을 경우, 해당 job의 상세 내역을 출력.

4.2.1.2. job compare

job에는 여러개의 target들이 등록될 수 있다. 해당 target들에 배포된 파일들이 정확히 동일한 파일들인지 이 명령어를 통해 확인할 수 있다.

```
job compare <job> [ -t <target> ... ] [ -failonly ]
```

해당 job에 속하는 target 중, 정합성 비교를 특정 target만 하고 싶다면 -t옵션 다음에 해당 target들의 이름을 스페이스로 구분지어 나열하면 된다. target에 배포된 파일이 많을 경우 -failonly 옵션을 붙이게 되면 동일하지 않은 파일들의 목록만 출력되기 때문에, 정합성 확인이 용이하다.

4.2.1.3. job mapping-scan

변경된 파일만 배포하려면 배포할 파일의 원본 파일이 필요하다. 다시말하면, 배포파일의 경우 일반적으로 빌드되어 나온 결과이기 때문에, 빌드하기전의 .java파일들의 정보가 필요한데, 이 정보를 job의 SOURCE설정시 Mapping에 추가해 주어야 한다. 이 명령어를 사용하면 현재 job에 알맞는 Mapping정보를 보여준다. 이 정보를 바탕으로 실제 job에 Mapping정보를 추가해 주어야 한다.

```
job mapping-scan <job>
```

명령어로 ODEN을 사용할 경우, xml을 직접 편집하여 job을 추가해 주어야 한다. ODEN Server의 conf폴더에 jobs.xml을 생성하여 아래와 같이 job을 추가할 수 있다.

```
<oden>
  <job name="batch" group="">
    <source dir="/home/oden/workspace/src"/>
    <target address="localhost:9872" dir="/home/oden/app0"
name="t0"/>
    <target address="localhost:9873" dir="/home/oden/app1"
name="t1"/>
  </job>
</oden>
```

가장 기본적인 형태는 위와 같다. oden태그로 시작하여 그 안에 여러개의 job을 설정할 수 있다. 하나의 job은 name이라는 attribute로 이름을 지정해 주어야 한다. job은 하나의 source를 가져야만 한다. job은 하나 이상의 target을 가질 수 있다. source는 ODEN Server가 설치된

시스템이어야 하며 해당 경로에 배포할 파일이 존재하게 될 것이다. target은 파일이 배포될 원격의 경로이며, ODEN Agent가 설치되어 있어야 한다. address에는 해당 시스템의 ip와 port(Agent는 기본으로 9872포트를 사용한다)를 적어 주어야 한다. dir에는 source의 파일이 배포될 경로를 지정해 주어야 한다. name은 이 target의 이름이며 필수 값이다. 같은 job안에서 target이름은 중복되어서는 안된다. 다른 job의 target이름과 중복되는 것은 허용된다.

```
<oden>
  <job name="batch" group="">
    <source dir="/home/oden/mypjt-1.0/dist"
excludes="**/*.svn">
      <mapping checkout-dir="/home/oden/mypjt-1.0" dir="."/>
      <mapping checkout-dir="/home/oden/mypjt-
1.0/src/main/java" dir="WEB-INF/classes"/>
      <mapping checkout-dir="/home/oden/mypjt-
1.0/src/main/resources" dir="WEB-INF/classes"/>
    </source>
    <target address="localhost:9872" dir="/home/oden/dest1"
name="t1"/>
    <target address="localhost:9873" dir="/home/oden/dest0"
name="t0"/>
    <target address="localhost:9974" dir="/home/oden/dest2"
name="t2"/>
    <command command="ls" dir="./" name="ls"/>
    <command command="./catalina.sh run"
dir="/Applications/tomcat-6.0.2/bin" name="tomcat"/>
    <command command="ps -ef | grep tomcat" dir="./"
name="ps"/>
  </job>
</oden>
```

위 예는 가장 복잡한 job설정의 예이다. 마찬가지로 job name이 존재하며, 하나의 source와 여러개의 target 이 존재한다. 그리고 여러개의 command가 존재하는 데 이것은 job에 속한 target에 명령을 내리고 싶은 경우, 쓰이게 된다. source에는 excludes라는 필드가 추가되었다. dir이하의 모든 파일 중 제외하고 싶은 목록을 스페이스로 구분하여 나열하면 된다. source하위에 mapping태그가 추가되었다. mapping은 변경된 파일만 배포하기 위해 추가적으로 지정해 주어야 하는 정보이다. class 파일 배포 시 빌드 되기 전의 java파일의 정보를 이용하여 변경된 파일을 감지하게 된다. checkout-dir에는 java파일이 존재하는 경로, dir에는 class파일이 존재하는 경로를 적어주면 된다. Anyframe으로 프로젝트를 진행하는 경우, job mapping-scan 명령어를 이용하면 어떤 정보를 넣어야 되는지 자동으로 찾아준다. command는 target의 시스템에 특정 명령을 실행하고 싶은 경우 지정해 주면 된다. name에는 명령어를 지칭할 이름을 적어주면 되며, command attribute에는 실제로 실행할 명령어, dir에는 어느 경로에서 command를 실행할 것인지를 적어주면 된다. 이 command 태그는 exec 명령어와 연관되어 사용된다.

4.2.2. Deploy Command

4.2.2.1. deploy test

배포될 파일의 목록을 미리 조회하는 명령어

```
deploy test <job> [ -t <target> ... ]
[-u | -i ] [ -del ]
```

- job의 target 중 특정 target에만 배포하고 싶다면, -t옵션 뒤에 target이름들을 스페이스로 구분하여 나열하면 된다.
- job의 SOURCE의 모든 파일을 배포하고 싶은 경우 -i 옵션을 지정하면 된다.
- job의 SOURCE의 파일 중 변경된 파일만 배포하고 싶다면 -u 옵션을 사용하면 된다. -i 옵션과 동시에 쓸 수 없다.
- target에 배포된 파일 중 job의 SOURCE에 존재하지 않는 파일을 삭제하고 싶을 경우 -del옵션을 사용하면 된다.

4.2.2.2. deploy run

SOURCE의 파일을 배포하는 명령어

```
deploy run <job> [ -t <target> ... ]
[-u | -i ] [ -del ] [ -c ]
```

- job의 target 중 특정 target에만 배포하고 싶다면, -t옵션 뒤에 target이름들을 스페이스로 구분하여 나열하면 된다.
- job의 SOURCE의 모든 파일을 배포하고 싶은 경우 -i 옵션을 지정하면 된다.
- job의 SOURCE의 파일 중 변경된 파일만 배포하고 싶다면 -u 옵션을 사용하면 된다. -i 옵션과 동시에 쓸 수 없다.
- target에 배포된 파일 중 job의 SOURCE에 존재하지 않는 파일을 삭제하고 싶을 경우 -del옵션을 사용하면 된다.
- job의 SOURCE의 파일을 압축 전송을 하고 싶은 경우 -c 옵션을 사용하면 된다.
- 배포 옵션을 지정하지 않을 경우 기본적으로 -i 옵션이 지정된다.

4.2.2.3. deploy runs

SOURCE의 파일을 배포를 일괄 처리하는 명령어.배포 작업을 멀티 스레드 처리

```
deploy runs <job> ...
[-u | -i ] [ -del ] [ -c ]
```

- job의 SOURCE의 모든 파일을 배포하고 싶은 경우 -i 옵션을 지정하면 된다.
- job의 SOURCE의 파일 중 변경된 파일만 배포하고 싶다면 -u 옵션을 사용하면 된다. -i 옵션과 동시에 쓸 수 없다.

- target에 배포된 파일 중 job의 SOURCE에 존재하지 않는 파일을 삭제하고 싶을 경우 -del옵션을 사용하면 된다.
- job의 SOURCE의 파일을 압축 전송을 하고 싶은 경우 -c 옵션을 사용하면 된다.
- 배포 옵션을 지정하지 않을 경우 기본적으로 -i 옵션이 지정된다.

```
ODEN Admin에서는 runs 사용이 불가능하다.
```

4.2.2.4. deploy rerun

배포 작업 후 예외가 발생한 작업만 재배포 하는 명령어

```
deploy rerun <txid>
```

- txid는 배포 작업 후의 유일한 값이며 txid를 통해 예외가 발생한 작업만 배포 작업을 재수행 한다. txid는 Log Command를 통해 확인 할 수 있다.

4.2.2.5. deploy undo

배포를 원복하는 명령어

```
deploy undo <txid>
```

- txid는 배포 작업 후의 유일한 값이며 txid를 통해 배포 작업을 원복한다. txid는 Log Command를 통해 확인 할 수 있다.

4.2.3. Log Command

4.2.3.1. log show

배포한 상세 내역을 보여주는 명령어.

```
log show <txid> [ -mode <A | U | D> ] [-path <path>] [-failonly]
```

txid에 해당하는 배포 내역의 상세를 보여준다. 새로 추가 되거나(A), 변경되거나(U), 삭제된(D) 파일의 목록만 보고싶다면 -mode옵션을 사용하면 된다. 배포된 내역중 특정 파일의 내역을 찾고 싶다면 -path 이하에 파일명을 입력하면 된다. 실패한 파일 목록만 보고자 한다면 -failonly 옵션을 덧붙여 명령을 실행하면 된다.

4.2.3.2. log search

배포된 목록을 간략히 보여준다.

```
log search [-job <job>] [-user <user>] [-path <path>] [-failonly]
```

특정 job의 배포내역을 보고 싶다면 -job옵션을 사용하면 된다. 특정 user가 배포한 배포내역을 보고자 한다면 -user 옵션을 사용하면 된다. 특정파일이 배포된 내역을 보고자 한다면 -path 옵션을 사용하면 된다. 실패한 목록만 보고자 한다면 -failonly옵션을 추가하면 된다.

4.2.3.3. log error

특정 날짜의 시스템 로그를 보여준다. 메시지만으로 배포가 왜 실패하였는지 알기 힘들 때, 혹은 개발자의 디버깅 용도로 사용된다.

```
log error [-date <date>]
```

-date옵션을 지정하지 않았을 경우 최신 시스템 로그를 출력한다. 시스템 로그는 날짜별로 oden 서버의 meta 폴더에 저장된다. 로그 파일의 크기가 10메가가 넘을 경우 기존 로그는 백업하고 새 로그파일에 로그를 기록한다.

배포 이력은 디폴트로 365일 동안 보관된다. oden.ini의 log.duration 속성을 통해 변경할 수 있다. 단위는 day이다.

4.2.4. Exec Command

4.2.4.1. exec run

exec명령어를 통해 job에 등록된 command를 실행시킬 수 있다. 해당 command 는 특별히 target을 지칭하지 않는 이상 job에 등록된 모든 Target을 대상으로 수행 된다.

```
exec run <job> [ -t <target> ... ] -c <command-name>...
```

job에 등록된 모든 target이 아닌 특정 target으로 한정짓고 싶다면 -t옵션을 사용하면 된다. job에 등록된 명령어 중 수행할 명령어의 이름을 -c옵션 이하에 나열한다.

4.2.5. Status Command

진행중인 배포내역을 확인하는 명령어이다. (Log 명령어는 기 배포된 내역과 관련된 명령어 이며, Status 명령어는 현재 진행중인 배포 내역과 관련된 명령어이다.)

4.2.5.1. status info

현재 진행중인 작업의 진행현황 혹은 대기 중인 작업 리스트를 출력한다.

```
status info
```

4.2.5.2. status stop

현재 진행중이거나 대기중인 배포 작업을 취소한다.

```
status stop <txid>
```

4.2.6. Build Command

4.2.6.1. build info

Jenkins의 등록되어 있는 job을 확인하기 위한 명령어.

```
build info <job>
```

job이름이 없는 경우 등록되어 있는 job 목록을 출력, job이름을 인자로 주었을 경우, 해당 job의 상세 내역을 출력.

4.2.6.2. build run

Jenkins 빌드를 실행하는 명령어.

```
build run [ <job> ]
```

4.2.6.3. build log

해당 job의 최근 수행된 빌드 정보를 확인하기 위한 명령어.

```
build log <job>
```

4.2.6.4. build status

해당 job의 진행현황을 확인하기 위한 명령어.

```
build status [ <job> ]
```

4.2.6.5. build check

Jenkins서버의 사용가능여부를 확인하기 위한 명령어.

```
build check
```

Chapter 5. Working with ODEN Admin

본 장에서는 ODEN의 Web 기반 GUI 환경에 대한 보다 상세한 이해를 위한 설명을 제공한다.

5.1. Introduction to ODEN Admin

ODEN Admin은 ODEN Server와 연계하여 배포관리를 지원하는 UI이며, 다음과 같은 주요 특장점을 가진다.

- Job 추가, 삭제 및 관리 기능 제공.
- Job Grouping 기능 제공.
- 배포할 파일 미리보기 기능 제공.
- 특정 서버에 대해 스크립트 실행 기능 제공.
- 서버간 배포 결과 비교 기능 제공.
- 배포 진행 상황 확인 기능 제공.
- 배포 결과 확인 기능 제공.
- 시스템 에러로그 확인 기능 제공.
- 사용자 권한 관리
- Jenkins 연동을 통한 빌드수행 기능 제공.

5.2. Job

Job은 어디에서 어디로 배포될 것인지에 대한 내용이 기술된 ODEN의 배포 설정 단위이다. Job은 하나의 SOURCE와 여러개의 Target으로 이루어져 있다. SOURCE는 배포할 파일이 위치한 경로(빌드서버)를 지칭하며, Target은 파일이 배포될 경로(개발 혹은 운영서버)를 말한다.

5.2.1. Job List

Job 목록 화면은 현재 등록된 Job들의 목록을 보여준다. Job Name 항목은 저장된 Job의 이름을 보여주며, 다른 Job 이름과 중복되어서는 안된다. 해당 이름을 클릭하면 Job 수정화면으로 전환된다. Job의 Status 항목은 해당 Job의 배포가 최근에 성공했는지 여부를 보여준다. 녹색이면 성공, 레드이면 실패다. Job의 Action 항목은 Job과 관련된 동작들을 수행할 수 있는 아이콘들을 보여준다. Action항목의 각 아이콘별 동작은 아래 표와 같다.

[[cols="10%,20%,70% ", options="header"]]

Icon	Name	Description

[job build]	Job Build	연동되어있는 Jenkins의 job 빌드시 일반적으로 사용하게될 버튼이다. 수행 후 나타나는 모니터버튼을 누르면 jenkins화면의 실행로그를 확인할 수 있다.
[job deploy]	Job Deploy	배포 시 일반적으로 사용하게될 버튼이다. 배포 대상이 되는 파일들을 미리 확인할 수 있다. 배포 대상은 지난 배포 시점이후 변경된 파일들이 될 수도 있고, 이미 배포된 파일중 삭제하여야 하는 파일이 될 수도 있다.
[job cleandeploy]	Job Clean Deploy	대부분의 경우 위의 Deploy를 사용하겠지만, 그것만으로 부족한 경우, 이 기능을 사용하면 배포되었던 모든 파일들을 삭제하고 전부 다시 배포하게 된다.
[job compare]	Compare Targets	Target Server가 여러대 인 경우 해당 서버들에 배포된 파일들이 정말 동일한 파일들인지 알기를 원할 때가 있다. 이 기능은 Target Server에 있는 파일들을 비교하여 정말 동일한 파일인지 알려주는 기능이다. Target Server가 ☒ 대 이상일 경우에만 사용할 수 있다.
[job runscript]	Run Script	Target Server에서 특정 명령어나 스크립트를 구동하고 싶은 경우 이 명령을 사용할 수 있다. 일반적으로 배포 전 서버 중지, 배포 후 서버 시작 등에 사용되며 이 기능을 사용하기 위해서는 수행할 명령어를 미리 Job의 Command 항목에 등록해 놓아야 한다.
[job del]	Delete Job	해당 Job을 삭제한다.
[job rollback]	Job Rollback	해당 Job의 최근 배포작업을 원복한다.

5.2.2. Add and Remove Job

Job 목록화면에서 우측 상단의 Add 버튼을 통해 새로운 Job을 등록할 수 있으며, 목록에서 기존에 있는 Job 이름을 선택하여 기존에 등록된 Job을 조회, 수정할 수도 있다. Action의 휴지통 아이콘을 클릭하면 해당 Job을 삭제할 수 있다.

5.2.2.1. Add Job

Job 목록화면에서 우측 상단의 Job 추가 버튼이나 기존 Job 이름을 클릭하면 Job 상세 화면으로 전환된다.

- Job Name

Job의 이름을 입력한다. Job을 구별할수 있는 유일한 키 이므로 기존의 Job 이름과 중복되지 않도록 한다.

배포할 파일(빌드서버)의 경로를 입력하는 란이다. ODEN Server가 빌드서버에 설치되므로 배포 파일이 존재하는 path만 절대 경로로 적어주면 된다. SOURCE의 항목 중 Directory 항목은 반드시 입력하여야 한다.

- Directory

배포할 파일들이 존재하는 경로중 가장 상위의 경로를 절대경로로 입력한다. 이 경로 이하의 폴더와 파일들이 그 구조 그대로 Target으로 전송되므로 이 디렉토리의 폴더 및 파일들을 미리 배포될 형태로 구성해 놓아야 한다.

```
ex> c:/anyframe/target(Windows)
    /anyframe/target(Unix)
```

Windows의 경우 폴더 구분시 역슬래쉬 사용이 가능하다.

- Excludes(optional)

배포 대상에서 제외할 파일에 대한 조건을 입력한다. 조건이 둘 이상일 경우 띄어쓰기를 통해 구분한다.

```
ex> **/*.jar **/.svn/**
    (모든 jar 파일과, .svn 폴더 하위의 모든 파일들 제외)
```

는 모든 하위폴더를 지칭한다. 는 현재 폴더만을 지칭한다. 예를들어, /.jar는 하위폴더의 모든 jar를 파일들을 제외하고 배포하겠음을 의미한다. /.svn/는 .svn이 들어가는 모든 폴더를 제외함을 의미한다.

- Mappings(optional)

Mapping은 변경된 파일만 배포하기 위해 설정해주어야 하는 부분이다. 예를들어 *.class 파일 배포 시, 빌드 전의 *.java 파일의 정보를 이용하여 변경 여부를 확인한다.

Anyframe프로젝트의 경우 Auto Mapping버튼을 이용하면 폴더구조를 분석하여 자동으로 mapping정보를 찾아준다. 단 기존의 mapping 정보가 모두 삭제되므로 주의한다. Auto Mapping을 이용하기 위해서는 Job이 서버에 등록되어 있어야 한다. 그러므로 Job에 대한 전체 설정을 마치고 Save를 한뒤 다시 Job으로 돌아와 Auto Mapping 버튼을 눌러주어야 한다.

일반적으로 빌드시 기존에 빌드되었던 리소스 및 class파일을 모두 삭제하고 전체를 새로 빌드하거나 복사를 하게 된다. 결국 변경된 파일뿐만이 아니라 모든 파일들이 새로 생성되게 되므로 모든 파일들의 날짜가 지금 시점으로 변경이 되어 버린다. ODEN에서는 파일의 timestamp를 이용하여 변경된 파일을 감지하게 되는데, 위와 같은 경우 모든 파일들의 날짜가 변경되었으므로 모든 파일이 변경되었다고 생각하게 된다. 하지만 빌드전 원본파일(class 파일의 원본파일은 java 파일임)의 경우 변경된 것만 날짜가 갱신되었기 때문에 이 정보를 알 수 있다면, ODEN에서는 어느 파일이 새로 변경된 것인지 판단할 수가 있다. 그래서 배포할 파일말고 그것의 원본파일의 경로를 적어주는 곳이 Mappings란 곳이다.(형상관리를 이용하여 java파일들 내려받을 경우에도 대부분의 경우 변경된 파일만 내려받기 때문에 변경되지 않은 파일들의 날짜는 예전 그대로 있게된다.)

이런 내용을 이해하기 힘들다면 단순히 Auto Mapping 버튼만 클릭해도 왠만한 mapping정보는 찾을 수 있다. Auto Mapping 은 WEB-INF폴더를 기준으로 mapping 정보들을 얻어내므로 배포 대상 폴더에 WEB-INF폴더가 없을 경우 아무 결과도 나타나지 않을 것이다.

일반적인 웹 어플리케이션의 형태를 보면 WEB-INF/classes 폴더에는 빌드된 class파일들이 있고, 우리는 그 파일을 Target Server로 배포하게 된다. WEB-INF/classes 폴더는 어딘간의 src파일이 빌드된 결과물이다. Mappings에는 WEB-INF/classes, src경로.. 이 두가지를 적어 주어야 한다.

- SUB DIRECTORY

SOURCE란의 Directory에 적었던 경로 이하의 상대 경로를 적어준다. WEB-INF/classes라고 적었다면 실제경로는 SOURCE Directory경로 + WEB-INF/classes가 될 것이다.

```
ex> WEB-INF/classes
```

- SCM MAPPING DIRECTORY

SUB DIRECTORY에는 배포될 파일이 있고 그것이 빌드나 복사가 되기전 실제 원본 파일이 있는 경로를 이곳에 적어준다. class 파일이 아닌 경우에도 새로이 복사가 일어난 파일이라면 복사전 원본파일의 경로를 이곳에 적어주어야 한다.

```
ex> c:/anyframe/src/main/java(Windows)
    /anyframe/src/main/java(Unix)
```

src폴더가 src/main/java와 src/main/resource 이렇게 두개가 있다면 이 두개의 빌드 결과 모두 WEB-INF/classes폴더로 가게 된다. 결국 mapping정보도 WEB-INF/classes - src/main/java, WEB-INF/classes - src/main/resource 이렇게 두개를 적어줘야 한다.

Targets

파일이 배포될 Target Server(개발 혹은 운영서버)를 입력하는 란이다. Target Server에는 ODEN이 Agent모드로 동작하고 있어야 한다. 하나 이상 필수로 입력해야만 한다.

- STATUS

Target Server를 모두 입력하고 저장한 뒤, Job을 다시 조회하면 STATUS 항목에 불이 들어온다. 녹색이면 Target Server의 ODEN Agent와 정상적으로 통신했다는 의미이고, 회색이면 대상서버의 ODEN Agent와 통신할 수 없음을 의미한다.

- NAME

Target Server를 지칭하는 이름을 지정한다. Job내의 다른 Target과 이름이 중복되어서는 안된다. 다른 Job의 Target과는 이름이 중복되어도 된다.

- URL

Target Server의 URL을 입력하는 란이다. ip:port 혹은 domain:port 형태로 입력하여야 한다. port는 ODEN Agent가 떠있는 포트로 기본 값은 9872이다.

```
ex> 127.0.0.1:9872
```

- PATH

Target Server의 어느 위치에 파일이 배포될 것인지 그 경로를 입력하는 란이다. 배포를 하게되면 SOURCE의 Directory이하의 파일 및 폴더들이 이 PATH이하로 그대로 전송이 된다.

```
ex> d:/anyframe/oden/server/webapp(Windows)
    /anyframe/oden/server/webapp(Unix)
```

Target이 ODEN Agent와 1:1로 맵핑 되는 것은 아니다. Target이 지칭하는 시스템에 ODEN Agent만 동작하고 있으면 되므로, 한대의 ODEN Agent와 여러개의 Target이 맵핑 될 수도 있다. 예를 들어 배포할 파일이 특정 시스템의 여러 경로에 배포되어야 할 경우, Target을 PATH만 다르게하여 여러개 등록할 수 있다.

Commands(optional)

Target에 지정한 시스템에 특정 명령이나 스크립트를 구동시킬 수 있다. 실행하고 싶은 명령어는 Commands항목에 미리 등록되어 있어야 한다. 등록된 명령어는 Job목록 화면에서 Run Script 아이콘을 클릭하여 실행시킬 수 있다.

- NAME

명령어를 지칭하는 이름이며, Job내의 다른 Commands 명과 중복되어서는 안된다.

- PATH

명령어가 수행될 위치를 입력한다. dir명령어를 수행할 경우 어느 위치에서 수행할 것인지 이곳에 입력해야 한다. Tomcat의 catalina.bat을 수행하고 싶다면 catalina.bat이 존재하는 경로를 입력하여야 한다.

```
ex> c:/anyframe/util/tomcat/bin(Windows)
    /anyframe/util/tomcat/bin(Unix)
```

- SCRIPT

실제 수행할 명령어나 스크립트를 입력한다. 명령어 뒤에 인자도 입력 가능하다. 명령어나 스크립트 입력시 ./startup.sh와 같이 앞에 ./같은 걸 붙여서는 안된다.

```
ex> startup.bat(Windows) startup.sh(Unix)
```

Warning

Job 등록시, Job Name과 Directory, Targets 1개이상 입력하여야 Job이 정상적으로 등록된다.

5.2.2.2. Build Job(optional)

연동되어 있는 Jenkins의 특정 job을 매핑시켜 빌드를 수행할 수 있다. Jenkins의 특정 job이 매핑되어 있어야 Job list화면의 빌드버튼이 활성화 된다.

- Job Name

매핑시키고자 하는 Jenkins의 job을 선택한다.

Warning

oden.ini의 build.url=""에 Jenkins url이 등록되어야 job list가 나타난다.

5.2.2.3. Delete Job

Job목록화면에서 삭제하고 싶은 Job의 휴지통 아이콘을 클릭하여 해당 Job을 삭제할 수 있다.

임의로 Job을 삭제하지 않기 위해, Admin role의 계정만 본 기능을 사용할 수 있다.

5.2.3. Job Deploy

배포 대상이 될 파일의 목록을 확인하고 그 리스트 중에서 특정파일만 배포 or 삭제할 수 있다. 배포 대상 파일 모두를 보려면 Deploy Scope의 All옵션을, 이전 배포 시점이후 변경된 파일만 보려면 Modified only를 선택한다. 배포된 파일중 Target에는 존재하나, SOURCE에 존재하지 않는 파일들을 보려면 Delete를 추가 체크한다.

- Modified Only

배포 대상 파일중 변경된 파일만 배포하고 싶은 경우 선택한다. Job설정시 Mapping정보가 정확히 설정되어 있지 않으면, 변경된 파일뿐만 아니라 변경되지 않은 파일까지 보이게 된다.

- All

배포 Directory 하위에 있는 파일 모두를 변경 여부에 상관없이 배포하는 대상으로 인식한다.

- Delete

Target과 SOURCE를 비교하여, Target에 불필요한 파일이 포함되어 있을 경우, 삭제를 할 것인지 무시할 것인지를 결정하는 옵션이다.

- Compress

배포 대상 파일을 압축하여 배포하고 싶은 경우 선택한다. 이는 네트워크 구간에서 속도 저하가 우려될 경우 사용한다.

배포 옵션을 선택한 후에는 Preview 버튼을 통해 배포 대상 리스트를 미리 확인할 수 있다. 배포 대상중 제외하고 싶은 항목이 있다면 해당 항목 우측의 X를 클릭하면 해당 항목은 배포 or 삭제가 되지 않을 것이다.

Deploy버튼은 현재 보여지는 페이지의 파일들만 배포하는 버튼이며, Deploy All은 모든 페이지의 파일들을 배포하는 버튼이다. 하지만 Deploy All버튼으로 배포 시 X표시를 눌러 특정 파일만 제외하고 배포하는 것이 불가능하다. 실제로 눌러졌더라도 배포는 전부 될 것이다. 특정 파일만 제외하고 배포 or 삭제하고 싶다면, Deploy버튼을 눌러 배포 or 삭제를 수행해야 한다. Deploy버튼을 눌러 수행하면, 화면 전환이 일어나지 않고 현재 화면에 머물러 있다가, 배포가 끝나는 시점에 화면에는 배포가 완료된 항목을 제외한 리스트를 다시 보여준다.

Deploy All 기능을 통해 배포를 진행하다보면 pop-up이 나타나는데, 이를 통해 배포 후에 해당 Job에 등록된 스크립트 명령어 중 어떠한 명령어를 실행시킬 것인가에 대해 설정할 수 있다. 전체적으로 배포를 진행한 후 스크립트를 동작시키기를 원하면, 해당 스크립트를 선택한 후 배포를 진행하면 된다. 만약 배포만 진행하길 원할 경우에는 None을 선택한 후 배포를 진행하면 배포 완료 후 어떠한 스크립트도 수행하지 않는다.

5.2.4. Job Clean Deploy

Target Server 디렉토리를 초기화하고, 현재 ODEN Server의 상태로 동기화 시키기 위한 명령이다. 다시 말하면, Target Server의 모든 파일이 삭제되고 SOURCE의 모든 파일이 다시 해당 서버로 배포가 되는 명령이다. 단, Deploy와는 다르게 미리보기를 지원하지 않으며 옵션을 선택할 수도 없다. 아이콘을 클릭하는 순간 바로 배포가 되므로 주의해야 한다.

5.2.5. Compare Targets

Target Server간의 정합성 확인을 위한 화면이다. Target들에 배포된 파일들이 서로 동일한 파일인지 아닌지를 검사하여 결과를 보여준다. 기본적으로 동일하지 않은 목록만 보여주며, 우측상단의 Failed Only버튼의 체크를 해제하면, 동일한 파일까지 포함된 Target들에 배포된 모든 파일들을 보여준다. 다음의 경우 동일하지 않은 파일로 나올 수 있다.

- 파일이 검사대상 Target중 일부 Target에 존재하지 않을 경우.

- 파일이 존재하나 파일의 사이즈가 틀린경우.
- 파일이 존재하나 파일의 Timestamp가 틀린 경우.

5.2.6. Run Script

Job설정시 등록한 Command를 수행할 수 있는 화면이다. 일반적으로 배포 전 서버 정지, 배포 후 서버 시작등의 명령을 수행하는 용도로 쓰인다. Job설정시 최소 하나 이상의 Command를 등록하였어야 이 화면에 접근할 수 있다. 스크립트 실행화면에서는 먼저 명령어를 수행할 Target들을 선택한 후, 우측의 Command 목록에서 수행할 명령어의 화살표 아이콘을 클릭하면 명령어가 수행이 된다. Target마다 수행해야 할 스크립트 위치가 틀린 경우, Job설정시 Commands의 PATH항목에 절대경로가 아닌 상대경로를 입력하는 것도 해결 방법이 될 수 있다.

dir과 같은 명령어는 명령이 바로 종료가 되어 명령어 종료 시점에 결과를 보여준다. 하지만 어플리케이션을 구동하는 명령어 등과 같은 경우, 어플리케이션이 종료가 될 때까지 명령어가 종료되지 않기 때문에, ODEN에서는 15초간 진행된 결과만 화면에 보여주고 빠져나오고, 해당 프로세스는 시스템에서 백그라운드로 동작하게 된다. Unix의 ps 명령어나 Windows의 작업관리자 등을 통해 해당 프로세스를 확인할 수 있다. 대기시간 15초는 core/conf/oden.ini의 exec.timeout 속성에서 변경할 수 있다.

5.2.7. Job Rollback

최근 배포한 파일을 원복 하는 명령이다. Admin 화면에서는 최신 배포만 기능을 제공 하며 이전의 배포 작업의 원복을 원할 경우는 Web Console에서 deploy undo txid를 직접 입력하여 수행한다.

5.3. History

현재까지의 배포 목록과 함께 작업 결과를 확인할 수 있는 화면이다.

5.3.1. Searching Histories

이제껏 배포한 이력들을 조회할 수 있다. 매 배포마다 고유의 ID가 부여되며, 첫번째 컬럼에서 확인할 수 있다. 컬럼 순으로 배포 ID와 배포성공 여부(그린이면 성공. 레드면 실패), 이 배포작업을 수행한 Job 이름, 배포 작업 일시, 배포한 파일 개수(성공한 파일 개수/전체 배포하기로 되어 있던 파일 개수), 해당 배포작업을 수행한 User의 Id순으로 보여진다.

배포 Id를 클릭하면 해당 배포의 상세 내역을 확인할 수 있다. 주로 어떤 파일이 배포되었고, 어떤 파일이 배포되지 않았는지 확인할 수 있다.

검색란을 통해 특정 조건에 맞는 배포작업만 확인할 수 있다. File에 배포한 파일명을 입력을 하면 해당 파일을 배포한 모든 배포작업 목록만 보여진다. File뿐만이 아니라 User Id, Job으로도 검색 가능하다. Failed Only가 체크가 되어있으면 모든 배포 수행 작업 중 실패한 배포작업 목록만 출력되게 된다. 디폴트로 실패한 배포 목록만 출력되므로 전체 배포 작업 목록을 보고 싶을 경우 Failed Only항목의 체크를 해제하고 검색하여야 한다.

배포 이력은 디폴트로 365일 동안 보관된다. core/conf/oden.ini의 log.duration 속성을 통해 변경할 수 있다. 단위는 day이다.

5.3.2. Searching Deploy File List

특정 배포 작업의 상세 정보를 확인할 수 있다. 왼쪽 컬럼부터 순서대로 인덱스, 해당 파일 배포 성공여부(그린이면 성공. 레드면 실패), Target Server명, 배포 파일명, 배포 모드(기존에 없는 파일을 새로 배포하게 되면 Add, 기존에 존재하던 파일을 새로운 파일로 덮어쓸 경우 Update, Target Server의 파일을 삭제할 경우 Delete), 그리고 배포 실패시 보여지는 Error Message가 있다.

배포파일 명으로 검색이 가능하며, 배포 모드별로 필터링도 가능하다. Failed Only에 체크가 되어 있으면, 배포에 실패한 아이템만 검색할 수 있다.

5.4. Status

진행중인 배포 작업과, 대기중인 작업 리스트를 모니터링할 수 있는 화면이다.

5.4.1. Job Status

현재 진행중이거나 대기중인 배포 작업들을 확인할 수 있다. 배포는 한번에 한 작업씩만 진행되므로 나머지 작업들은 대기중인 상태를 확인할 수 있다. 진행중인 작업을 중단하거나 대기중인 작업을 취소할 수 있다.

현재 진행중인 배포 작업에 대해서는 작업 진행률과 배포중인 파일에 대한 정보가 나타난다.

5.5. Log

ODEN Server에 발생한 System Log를 확인할 수 있는 화면이다.

5.5.1. ODEN System Log

배포 시 문제가 발생했을 경우, 그 원인을 찾는데 메시지만으로 찾기 힘들때 도움이 되거나, 개발자의 디버깅 용도로 사용하기위한 화면이다. Log는 각 날짜별로 확인할 수 있으며, 이를 통해 ODEN Admin과 ODEN Server 사이에서 어떤 문제가 발생했는지 확인할 수 있다.

Admin role의 계정만 본 기능을 사용할 수 있다.

5.6. User

배포 작업을 수행할 수 있는 사용자를 등록하여 권한을 부여하고, 주어진 권한을 제어하며, 등록된 사용자를 삭제할 수 있는 기능을 가진 화면이다. 사용자에 관한 내용을 제어하므로 권한을 가진 일부 사용자만 접근할 수 있다.

Admin role의 계정만 본 기능을 사용할 수 있다.

5.6.1. User List

User 목록화면은 현재 등록된 사용자들의 목록을 보여준다. Role이 Deployer인 사용자에게 주어지는 권한은 Job별로 주어지며, 권한이 있는 Job에 대해서만 상세정보, 배포이력(History) 정보를 확인할 수 있다. Role이 Admin인 사용자는 모든 Job에 대한 권한이 주어지므로 모든 Job 정보를 확인할 수 있다.

만약 Role이 Admin인 사용자라면, 모든 Job에 대해 권한이 있으므로 Assigned Job List에는

All Jobs라고만 나타난다. 하지만 모든 Job에 대해 권한이 있는 사용자라고 해도 Role이 Deployer일 경우에는 Assigned Job List에는 모든 Job List가 나타난다.

5.6.2. Add User

사용자는 페이지 상단의 Add 버튼을 통해 추가할 수 있다. 표 하단에 사용자 정보를 입력할 수 있는 부분이 나타나며, 이미 등록된 사용자를 클릭하면 사용자의 정보를 확인할 수 있다. 사용자 정보를 확인하는 도중에 새로운 사용자를 추가하려면 Add 버튼을 눌러 보여지는 사용자 상세정보를 초기화 한 후에 입력하면 된다.

- User ID: 사용자가 사용할 ID
- Role Name: 등록할 사용자의 권한(Admin / Deployer)
  - Admin : 모든 Job에 접근이 가능하며, User 메뉴 접근이 가능
  - Deployer : assign된 Job에 대해서만 접근이 가능하며, User 메뉴에는 접근 불가
- Password / Confirm Password: 사용자가 사용할 비밀번호
- Assign Job: 사용자에게 접근 권한이 주어진 Job List(Job이 둘 이상일 경우 컴마(,)를 통해 구분한다.)

5.6.3. Remove User

User 목록에서 삭제하고 싶은 사용자 계정을 User의 휴지통 아이콘을 클릭하여 해당 사용자를 삭제할 수 있다.

계정 관리를 위해 기본 제공된 사용자 계정(oden)을 제외한 모든 계정은 삭제가 가능하다.