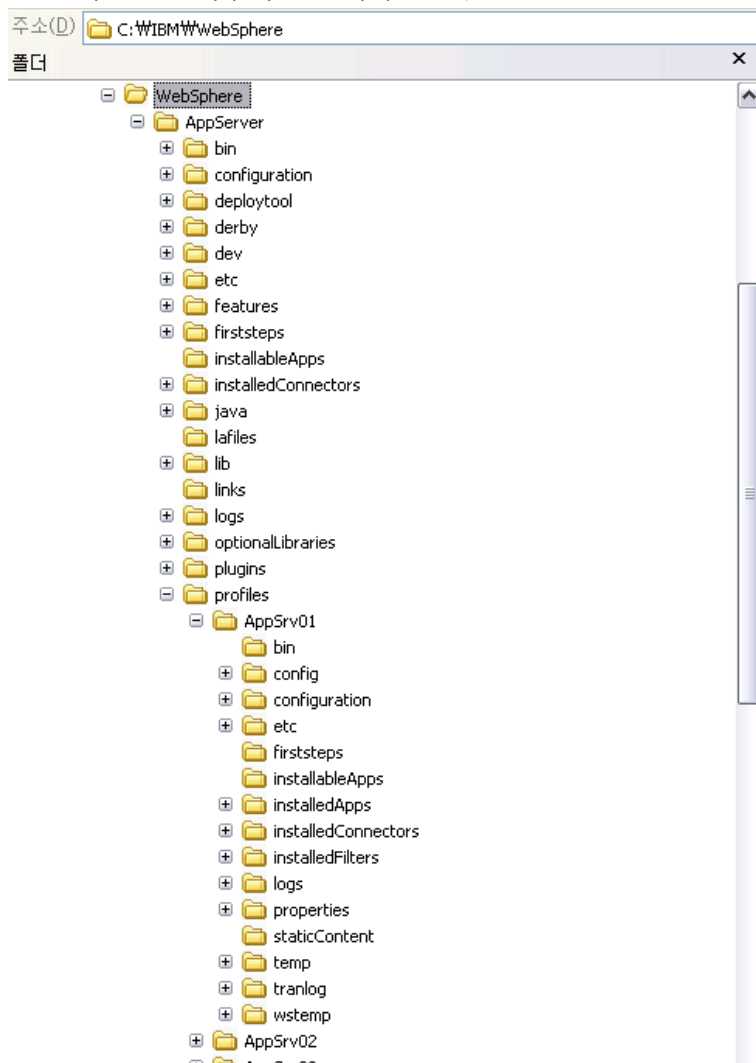


하나씩 쉽게 따라 해보는 IBM WebSphere Application Server(WAS) v7 – 17

이정운 (juwlee@kr.ibm.com)

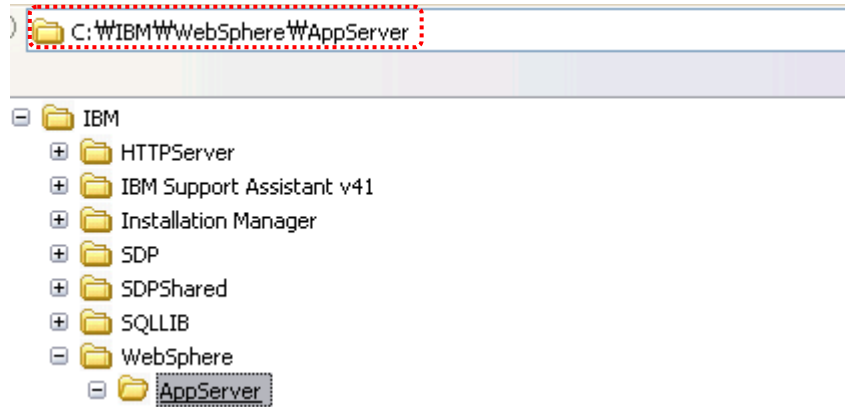
하나씩 쉽게 따라 해보는 IBM WAS v7 그 열 일곱번째 이야기를 시작합니다. 열 일곱번째라, 어느새 열일곱번째 강좌까지 왔습니다. 시간 참 빠르죠^^& 열 일곱번째 이야기는 지난 강좌들 하고는 조금 다르게 WAS 가 설치된 폴더 구조에서 알아야 할 부분이 무엇인지 이야기 할 것이며, 추가적으로 실제 설정들이 어떻게 되어 있는지, 어플리케이션을 배포하게 되면 무슨 일이 벌어지는지 등에 대한 조금은 세세한 이야기이긴 해도 반드시 알아두면 도움이 되는 이야기를 하려고 합니다. (그 동안은 WAS 의 기능 위주로 했지만, WAS를 잘 이해하기 위한 기초같지만 잊고 있었던 이런 부분도 계속해서 다루려고 합니다.) 단순 관리의 목적이라면 본 강좌를 Skip 하셔도 상관없지만 내부 구조에 대해 어떻게 구성되고 돌아가는지에 대한 이해는 앞으로 문제가 생겨서 대처해야 하거나 WAS 를 좀 더 자세히 이해하고자 할 때 무엇을 보고 어떻게 해야하는지에 대한 이야기를 해줄 수 있기 때문에 잘 보고 잘 습득하신 후에 넘어가시면 많은 도움이 될거라 생각합니다. (Windows 환경의 폴더를 보고 설명드리지만 Unix 나 다른 환경에서도 동일한 폴더 구조를 가지므로 걱정말고 봐두시면 됩니다.^^&)



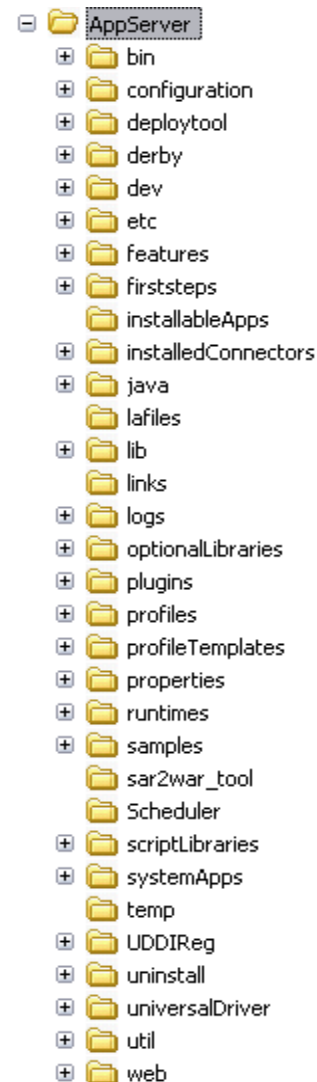
<그림 1> 기본적인 WAS 폴더 구조

Part 1. 기본적인 WAS 폴더 구조

기본적으로 강좌를 따라서 아무 변경없이 WAS 를 설치하셨다면 하단과 같은 폴더 구조를 볼 수 있으실 텐데, 여기서 가장 중요한 것은 C:\IBM\WWebSphere\AppServer 여기까지가 바로 **WAS_ROOT** 폴더라는 점 입니다. 기본적으로 WAS ROOT 폴더라고 이야기하는 것은 여기를 말한 다는 것을 숙지하시고 강좌를 이어가시면 됩니다.

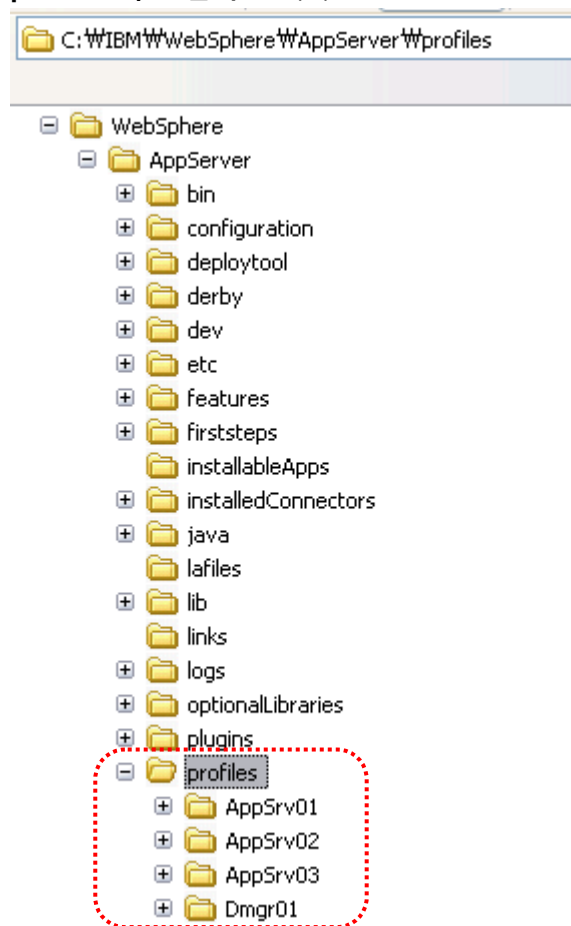


이 WAS ROOT 폴더 밑에는 하단에 보시는 것 처럼 아주 다양한 폴더들이 있습니다.



이 폴더의 역할들을 반드시 전부 다 외울 필요는 없지만 중요한 몇 몇 폴더의 역할은 이해하는 것이 좋습니다. 특히, 이전에 설명한 것처럼 IBM WebSphere 의 경우에는 profile 이라는 개념을 사용한다고 이야기 했습니다. Binary 파일이라는, 제품에 관련되어 모든 사용자가 공유하는 제품 파일(WAS 실제 엔진 이라고 보시면 됩니다.)이 있는 반면에 각 각의 사용자에게 맞게 customize 한 환경과 설정 파일들이 담긴 profile 이라는 것이 존재 합니다. Binary 파일이 지금 보시는 몸통과 같은 모든 디렉토리 폴더입니다. 그렇기 때문에 사실, 이 폴더들을 더 들어가 보거나 변경하실 일 들은 거의 일어나지 않기 때문에 여기서는 우선 패스하도록 하겠습니다.

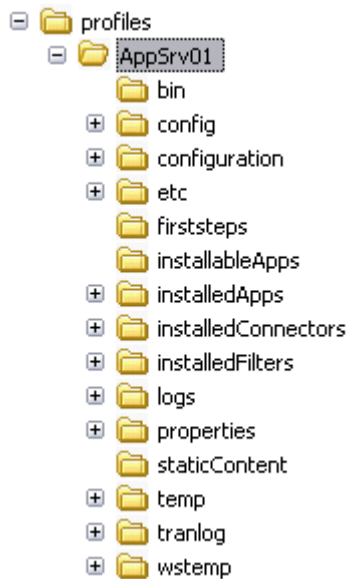
그러나 profile 을 가지고 있는 폴더는 다릅니다. 각 사용자에게 맞게 변경된 설정 파일들이 담겨 있기 때문에 필요에 의해서 가장 많이 확인하고 변경 할 수 있는 가능성이 있습니다. Profile 이 저장된 폴더가 어떤 것인지 아시겠나요? 너무 쉽죠..^^& 그 폴더가 바로 하단에서 보여지는 **profiles** 라는 폴더 입니다.



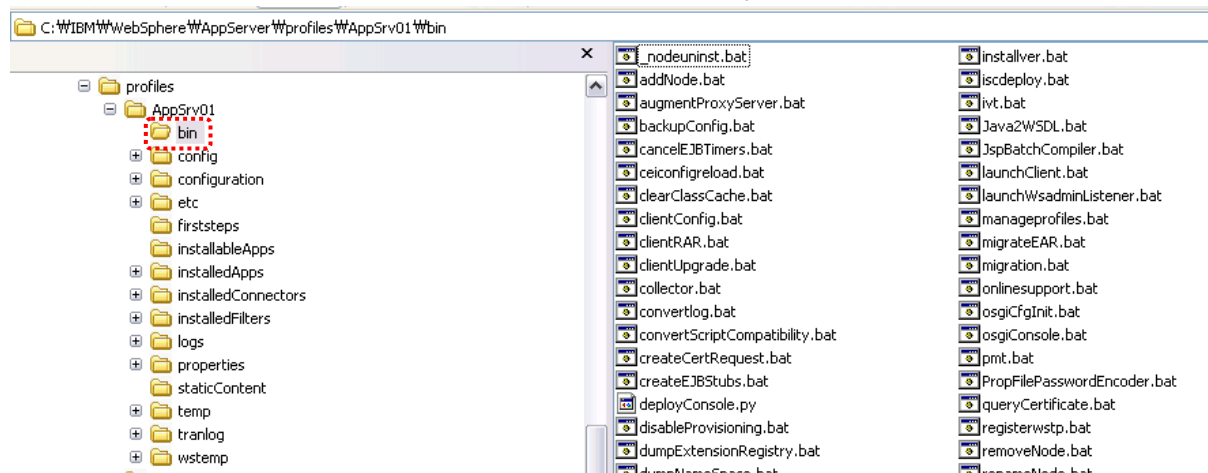
결국 profiles 하단의 폴더 구조로 제가 이전에 만든 profile 이 하나씩 볼 수 있고, 이 단위가 바로 이전에 설명한 node 단위와 동일 합니다. (제 컴퓨터의 경우에는 Dmgr 노드가 하나있으며, AppSrv01~03 까지 3개의 node 가 있는 구조 인 것입니다.)

그럼 이 profiles 폴더를 좀 더 들어가서 분석해 보도록 하겠습니다.

Profile 폴더를 클릭하여 내부 내용을 살펴보면 아래와 같은 구조를 가지고 있습니다. 이 구조는 어떤 node 이건 상관없이(Dmgr 노드이건, 일반 노드이건) 아래와 같은 구조를 가지고 있습니다.



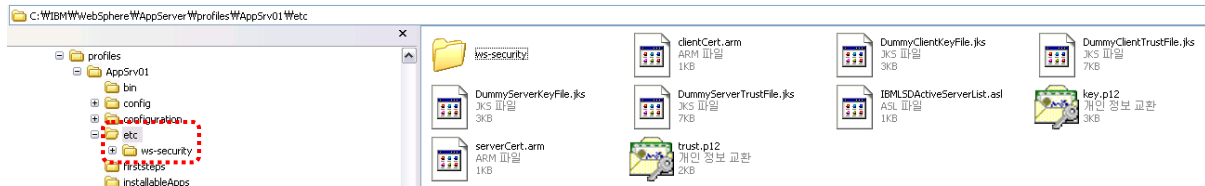
우선 **bin** 폴더를 보면 하단에 보시는 것처럼 각종 실행 script 들이 들어있습니다.



서버를 실제 실행할 수 있는 startServer.bat 스크립트 파일도 있고, 노드 에이전트를 실행하는 startNode.bat 이라는 실행 파일도 있습니다. 뿐만 아니라 설치된 WAS 버전을 확인할 수 있는 versionInfo.bat 라는 실행 파일 같은 각종 활용할 수 있는 스크립트 실행 파일들이 존재 합니다. (테스트로 이것 저것 인터넷에서 찾아보고 직접 실행해 보시는 것도 재미있을 것 입니다.)

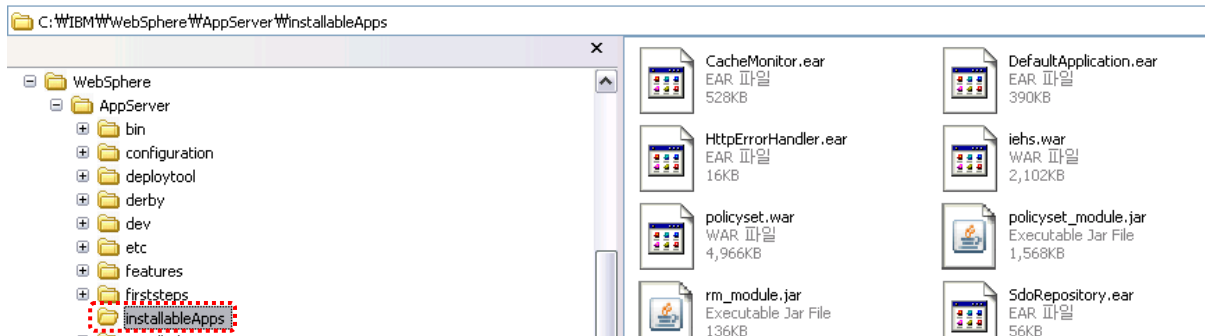
Config 와 configuration 은 실제로 설정이 들어있습니다. Configuration 폴더 내에는 eclipse 기반의 osgi 나 runtime 관련 설정이 있는데 변경하는 일이 없기 때문에 그냥 넘어가겠습니다. 보다 중요한 것은 실제 profile 에서 사용하는 설정들이 xml 형태로 들어가 있는 Config 폴더인데 많은 설명이 필요하므로 우선 skip 하고 넘어가도록 하고 뒤에서 다시 설명하겠습니다. (part 2 참조)

Etc 폴더는 보통 보안관련된 Key 값들과 Key DB 들이 들어가 있습니다. 보안을 걸거나 SSL 을 사용하면 여기에 있는 보안 관련 Key 를 보통 활용합니다.



Firststep 폴더는 설치하고 나서 나오는 firststep 이라는 처음 실행 마법사 같은것에 관련된 메뉴들이 들어가 있는 폴더입니다. 중요하지 않으므로 바로 넘어가겠습니다.

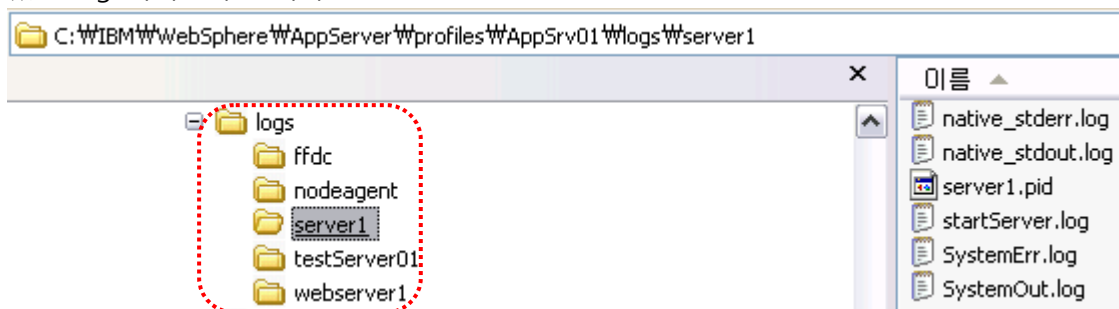
InstallableApps 폴더는 설치가 될 만한 유용한 Application 들이 있는 폴더 인데, profile 내에 있는 폴더에는 아무 것도 없는 빈 상태 입니다. 그러나 WAS ROOT 밑의 installableApps 폴더 내에는 하단에 보시는 것처럼 유용한 Application 들이 들어가 있습니다. 용어 그대로 필요하면 여기 있는 어플리케이션을 설치할 수 있으므로 설치해서 사용하면 됩니다.



InstalledApps 폴더는 해당 profile 로 설치된 어플리케이션이 기본적으로 설치되는 위치입니다. 이 폴더도 말할 것이 많으므로 뒤에서 다시 자세하게 설명하도록 하겠습니다. (part 3 참조)

InstalledConnectors 폴더와 **InstalledFilters** 폴더도 용어 그대로 설치된 Connector 와 Filter 가 설치되는 위치입니다. 보통 MQ connector 같은 경우에 InstalledConnectors 폴더에 설치되어 있습니다. Filter 는 보통 사용하지 않으니 넘어가도록 하겠습니다.

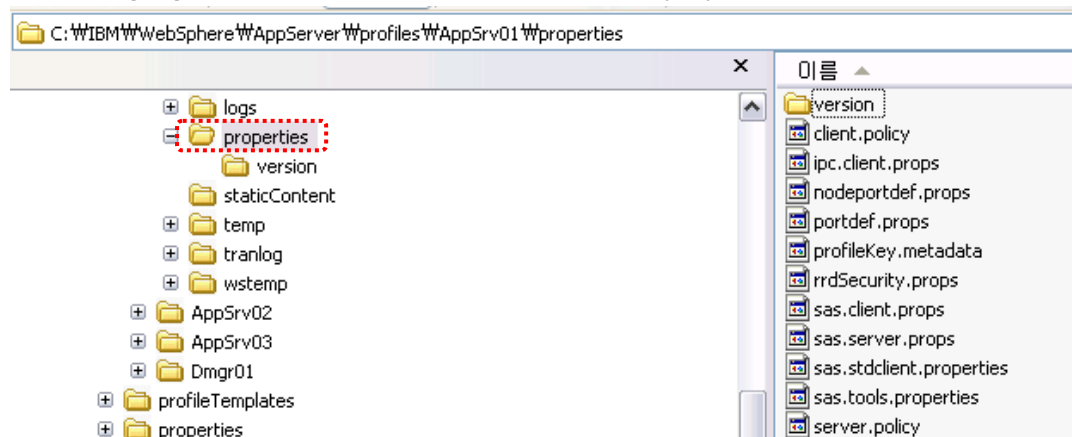
그 다음이 바로 이름만으로 용도를 알 수 있는 **logs** 폴더 입니다. 각 서버별로 log 가 저장되어 있는 log 디렉토리 입니다.



ffdc 라는 폴더가 잘 이해가 안가실 텐데 IBM 에서 분석을 위해서 사용되는 로그로서 FFDC (First Failure Data Capture) 라고 합니다. 원래는 AIX 에서 나온 개념을 WAS 같은 소프트웨어에 적용한 개념으로 장애시에 바로 Capture 된 data 들이 log 되는 위치입니다. IBM 에서 장애분석을 위해서 사용되는 log 입니다.

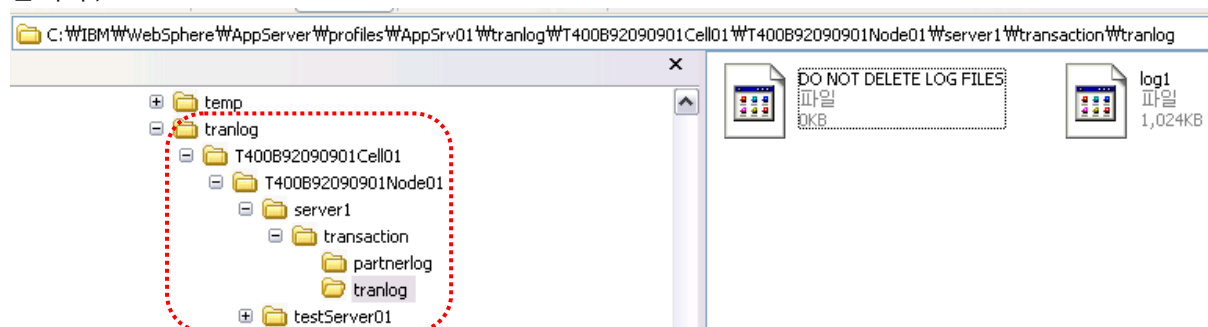
그러면 이제 각 서버에 있는 log 를 조금 더 자세하게 살펴보도록 하겠습니다. 우선 가장 중요하게 보실 것은 SystemOut.log 파일입니다. WAS에서 Output 으로 나오는 모든 log 가 기록되며 어플리케이션에서 System.out.println() 메소드가 수행되는 부분도 모두 이 SystemOut.log 에 기록되어 있습니다. Exception 도 보통 여기서 확인 가능하므로 가장 많이, 자주 보는 로그입니다. 그 다음이 SystemErr.log 인데 이름 그대로 Error 가 기록되는 로그입니다. startServer.log 와 stopServer.log 는 서버가 시작할 때와 중지될 때 기록되는 로그이며 native_stdout.log, native_stderr.log 는 OS 와 관련된 log를 기록해야 할 필요가 있을 때 사용되는 로그로서 직접 사용자가 살펴보는 경우는 극히 드문 일 입니다.

다음으로 **properties** 폴더는 WAS 에서 사용하는 각 properties 가 보관된 폴더 입니다.



Temp 와 **wstemp** 는 말 그대로 WAS 에서 사용하는 임시 temp 폴더 입니다.

마지막으로 **tranlog** 라는 폴더는 이름 그대로 트랜잭션 로그이며 트랜잭션에 대한 로그들이 쌓입니다. 해당 로그들은 장애에 의한 트랜잭션 복구를 위해서 사용됩니다. (장애에 의해서 서버가 죽었다가 다시 시작할 시에 트랜잭션 복구를 진행하는데 가끔씩, 이 것 때문에 오히려 시작이 안되는 경우에는 해당 파일을 삭제하시면 트랜잭션 복구 없이, 문제 없이 시작할 수 있는 경우가 있습니다.)



Part 2. Profile 의 config 폴더

이전에 이야기 했지만 내용이 많다고 넘어간 profile 폴더 밑의 config 폴더에 대해서 차근 차근 설명을 진행하도록 하겠습니다. Config 폴더에는 WAS 의 실제 설정들이 xml 형태로 저장되어 있습니다. 즉, 이전에 관리콘솔로 작업하는 것들을 많이 보여드렸는데 그 때 관련된 모든 설정은 사실적으로 이 Config 폴더 밑에 xml 파일 형식으로 저장되어 있습니다.

하단과 같이 관리 콘솔에서 확인하고 체크하고 변경할 수 있는 모든 옵션들은 관리 콘솔에서 하단처럼 보여집니다.

[Application servers](#) > [testServer01](#) > Session management

Use this page to configure session manager properties to control the behavior of Hypertext Transfer Protocol support. These settings apply to both the SIP container and the Web container.

Configuration

General Properties

Session tracking mechanism:

☐ Enable SSL ID tracking

☒ [Enable cookies](#)

☐ Enable URL rewriting

☐ Enable protocol switch rewriting

Maximum in-memory session count:
 sessions

☒ Allow overflow

Session timeout:

☐ No timeout

☒ Set timeout
 minutes

Additional Properties



- [Custom properties](#)
- [Distributed environment settings](#)

그러나, 실질적으로는 하단에 보는 것처럼 XML 파일 안에 해당 내용이 들어가 있으며, 이 파일의 내용이 변경되고 수정되면서 실제로 옵션이 변경되는 것입니다.

```
server.xml
0      10      20      30      40      50      60      70      80      90      100
1 <?xml version="1.0" encoding="UTF-8"?>
2 <process:Server xmi:version="2.0" xmlns:xmi="http://www.omg.org/XMI" xmlns:adminservice="http://www.ibm.com/xmlns/ibm-1.0"
3   <stateManagement xmi:id="StateManageable_1260773204390" initialState="START"/>
4   <statisticsProvider xmi:id="StatisticsProvider_1260773204390" specification="com.ibm.orb.enabled"/>
5   <services xmi:type="pmiservice:PMIService" xmi:id="PMIService_1260773204390" enable="true" initialSpecLe
6   <services xmi:type="adminservice:AdminService" xmi:id="AdminService_1260773204390" enable="true" standal
7     <connectors xmi:type="adminservice:SOAPConnector" xmi:id="SOAPConnector_1260773204390" enable="true">
8       <properties xmi:id="Property_1260773204390" name="requestTimeout" value="600"/>
9     </connectors>
10    <connectors xmi:type="adminservice:RMICConnector" xmi:id="RMICConnector_1260773204390" enable="true"/>
11    <connectors xmi:type="adminservice:JSR160RMICConnector" xmi:id="JSR160RMICConnector_1260773204390" enabl
12    <connectors xmi:type="adminservice:IPCConnector" xmi:id="IPCConnector_1260773204390" enable="true">
13      <properties xmi:id="Property_1260773204391" name="requestTimeout" value="600"/>
14    </connectors>
15    <configRepository xmi:id="RepositoryService_1260773204390"/>
16    <pluginConfigService xmi:id="PluginConfigService_1260773204390" enable="true"/>
17  </services>
```

팁 : 관리콘솔의 설정들이 어떤 XML 파일안에 들어있는지 확인이 어려운 경우에 알아볼 수 있는 손 쉬운 방법이 있습니다. 그건 바로 실제 설정의 XML 위치를 알고자 하는 설정을 직접 관리콘솔에서 변경하는 것 입니다. 다만, 변경 후에 저장을 누르지 않고 검토(review) 를 누르게 되면 해당 설정이 저장된 XML 파일을 직접 확인할 수 있습니다. (당연히 버리기(Discard) 로 해당 설정 변경은 반영하지 않으면 됩니다.)


Messages

-  Changes have been made to your local configuration. You can:
 - [Save](#) directly to the master configuration.
 - [Review](#) changes before saving or discarding.
-  The server may need to be restarted for these changes to take effect.

[Data sources](#) > **Save**

Save your workspace changes to the master configuration.

Click Save to update the master repository with your changes. Click Discard to discard your changes and begin work again using the master repository configuration. Click Cancel to continue working with your changes.

 Total changed documents: 1

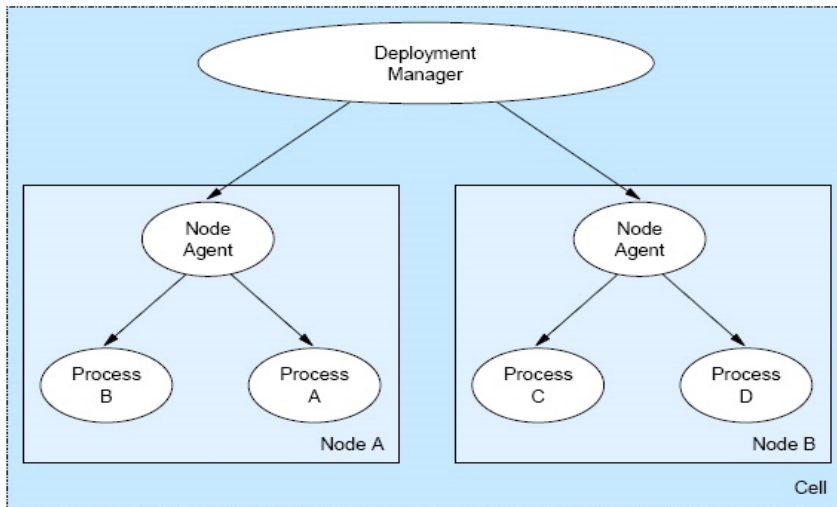
Changed Items	Status
cells / T400B92090901Node01Cell / nodes / T400B92090901Node03 / servers / server1 / resources.xml	Updated

[Save](#)

[Discard](#)

[Cancel](#)

이때, WAS ND 에서 설명했던 하단의 토폴로지 구조처럼 xml 설정 파일들을 보관하고 있습니다.
(그림은 뭔가 입체적으로 보일지 모르지만 결국은 tree 구조 입니다.)

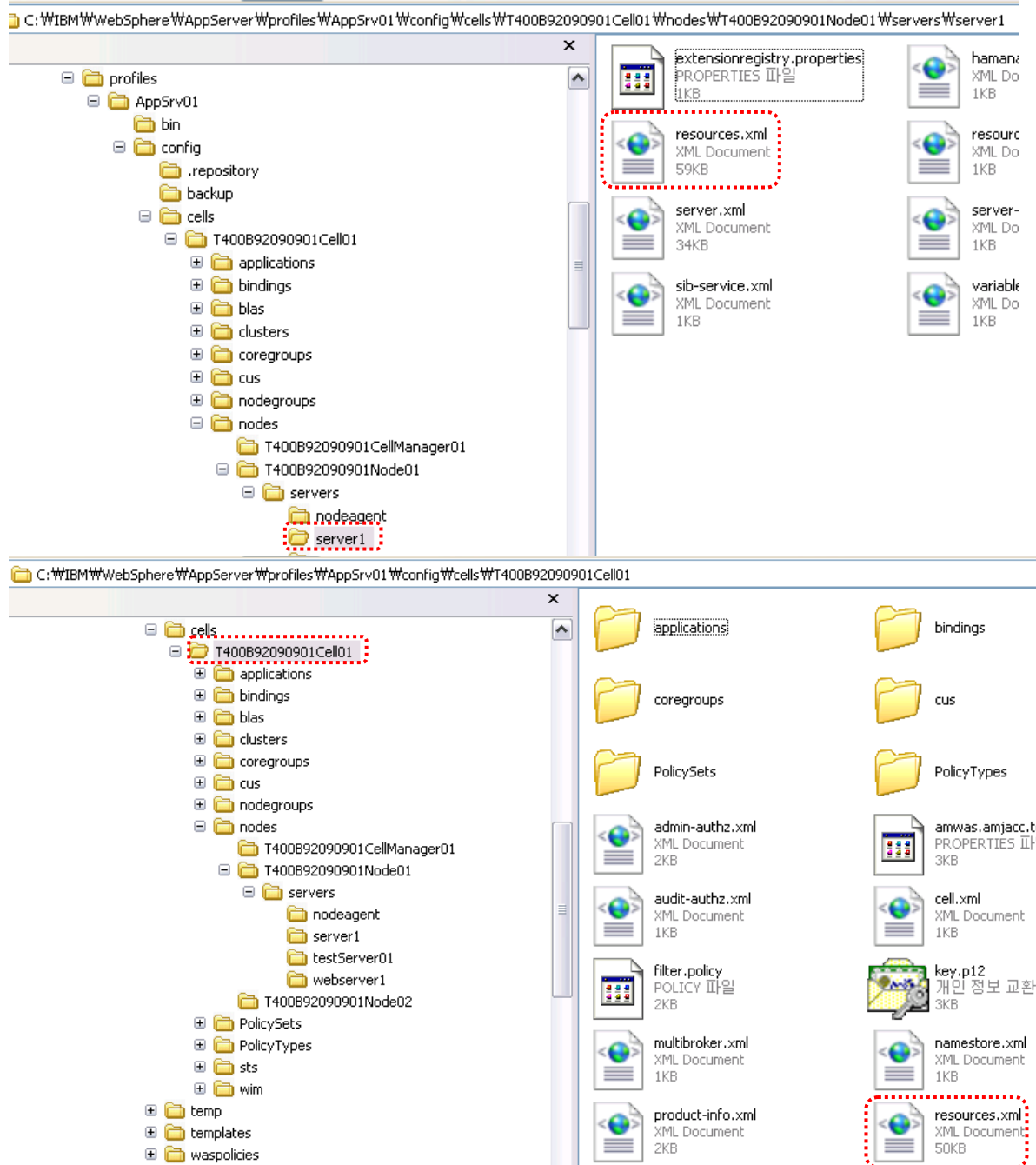


하단에 보시는 것처럼 WAS의 토폴로지를 생각하여, config 폴더 밑에 cells > 실제 Cell 이름 > nodes > 실제 노드의 이름 > servers > 실제 서버의 이름 폴더에 가면 하단처럼 Server 단위에서 필요한 xml 파일들을 확인할 수 있습니다. 이 파일들이 바로 WAS 설정이 저장되어 있는 xml 파일들입니다.

C:\IBM\WebSphere\AppServer\profiles\AppSrv01\config\cells\T400B92090901Cell01\nodes\T400B92090901Node01\servers\server1

이름	크기	종류
extensionregistry.properties	1KB	PROPERTIES 파일
hamanagerservice.xml	1KB	XML Document
libraries.xml	1KB	XML Document
pmi-config.xml	9KB	XML Document
resources.xml	59KB	XML Document
resources-cei.xml	1KB	XML Document
resources-pme502.xml	1KB	XML Document
resources-pme.xml	3KB	XML Document
server.xml	34KB	XML Document
server-cei.xml	1KB	XML Document
server-pme51.xml	1KB	XML Document
server-pme.xml	2KB	XML Document
sib-service.xml	1KB	XML Document
variables.xml	1KB	XML Document
ws-security.xml	16KB	XML Document

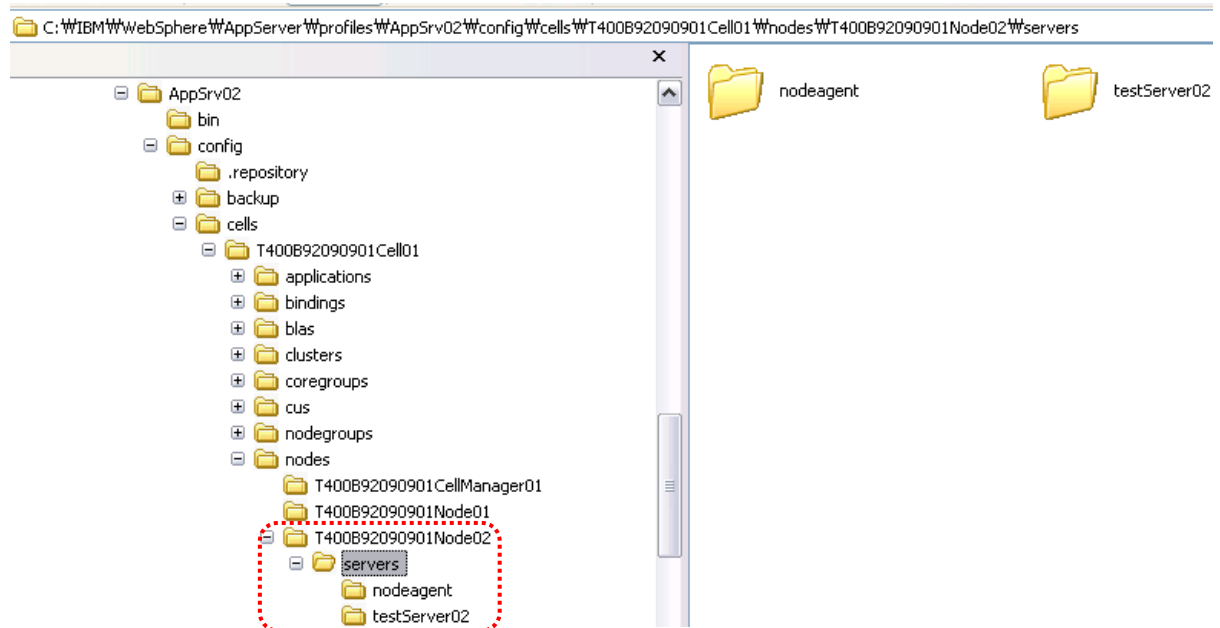
이 때, 좀 더 Config 폴더의 구조와 내용을 찬찬히 살펴보면 아시겠지만, 예를 들어 resources.xml 이라는 설정 파일의 경우는 server 밑에도 존재하고 node 밑에도 존재하고, 심지어 Cell 밑에도 존재하는 것을 하단처럼 확인할 수 있습니다.



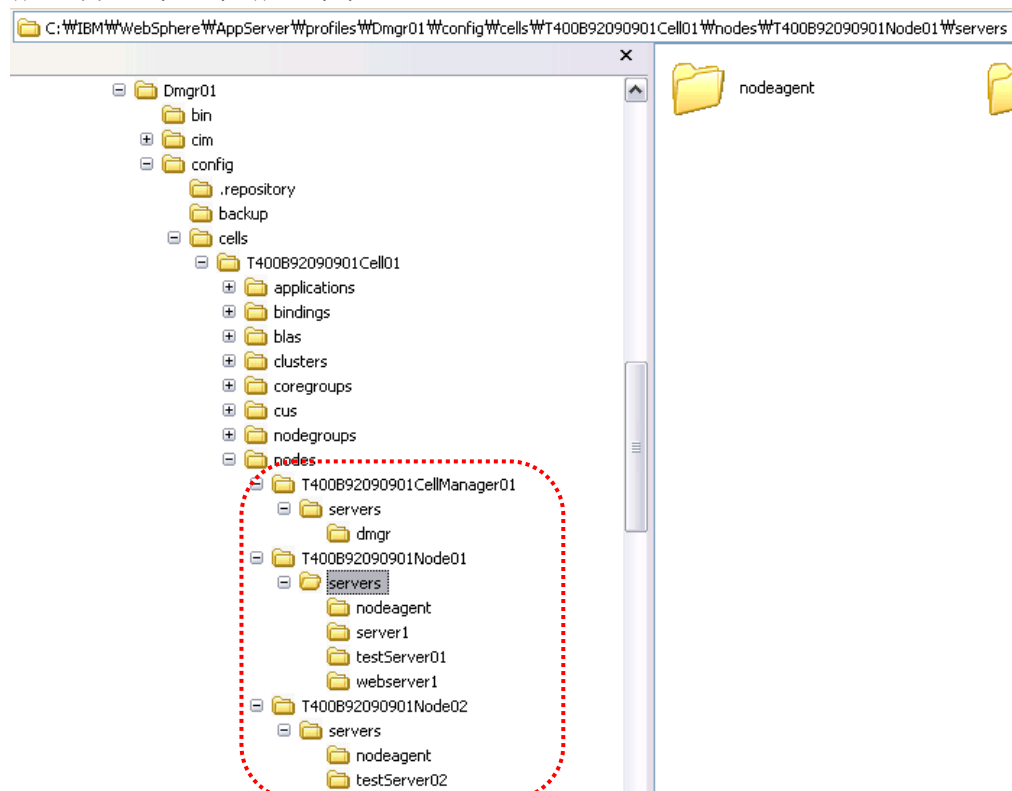
이렇게 중복되게 동일 파일이 WAS 토폴로지 구조대로 존재하는 것은 이전에 WAS 옵션을 설정할 때 이야기 했던 scope(범위) 과 설정들이 연결되기 때문입니다. WAS 를 설정할 때 scope 을 어떻게 지정하냐에 따라서 config 폴더 안의 어떤 위치에 있는 설정 파일이 변경되고 적용되는 것이 달라집니다. 좀 더 쉽게 예를 들면, Cell 레벨의 설정을 변경하면 Cell 폴더 밑에 있는 resources.xml 파일이 변경되며, Server 레벨의 설정을 변경하면 Server 폴더 밑에 있는 resources.xml 이 변경되는 것입니다. 위의 WAS 토폴로지 그림과 맞추어서 생각하시면 편하게 이

해하실 수 있습니다.

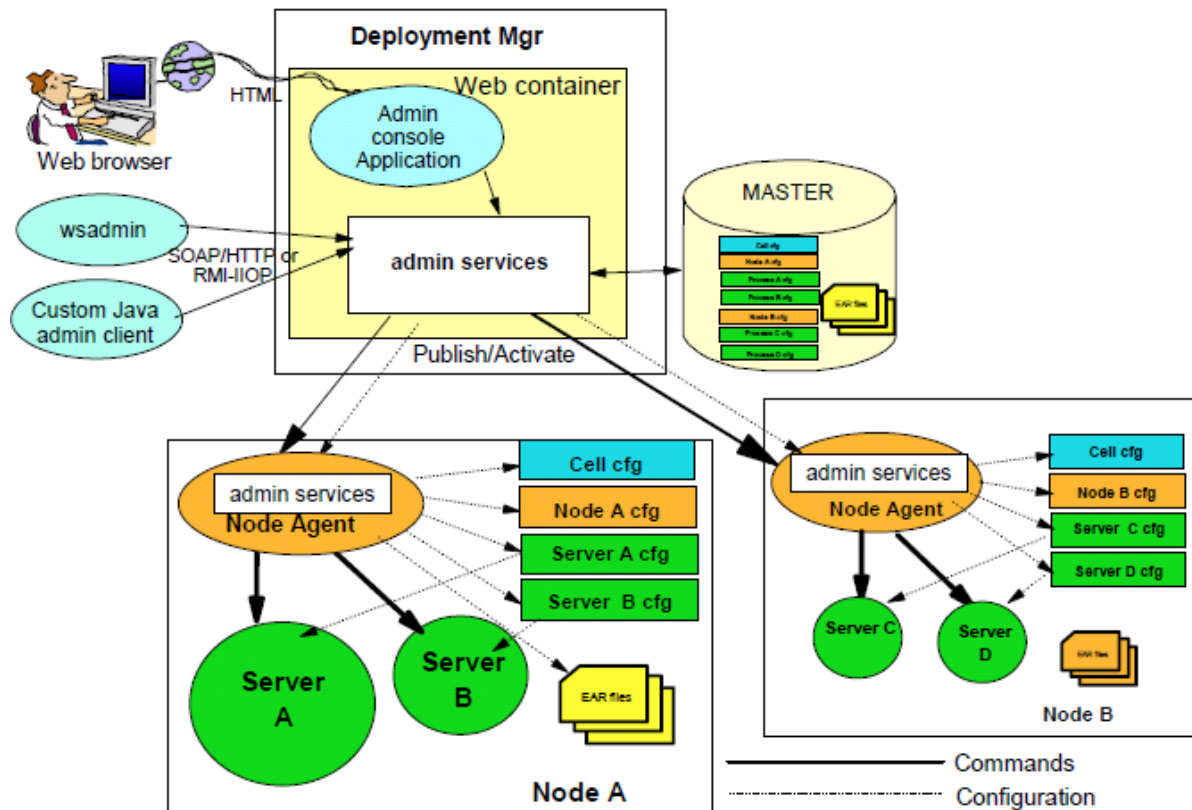
그렇다고 profile 에 모든 WAS 토폴로지의 설정이 들어있는 것은 아닙니다. profile 마다 WAS 의 토폴로지 구조대로 config 폴더에 해당 설정 파일을 보유하고 있습니다. 위의 경우에는 첫번째 node 밑의 설정들만 확인할 수 있는 것 처럼 하단처럼 두 번째 profile 의 config 를 보면 두번째 node 하단의 설정들만을 확인할 수 있는 것을 알 수 있습니다.



이와 다르게 Dmgr 밑에 있는 config 폴더의 설정들을 보면 모든 node 의 설정을 다 확인할 수 있는 것을 아실 수 있습니다.

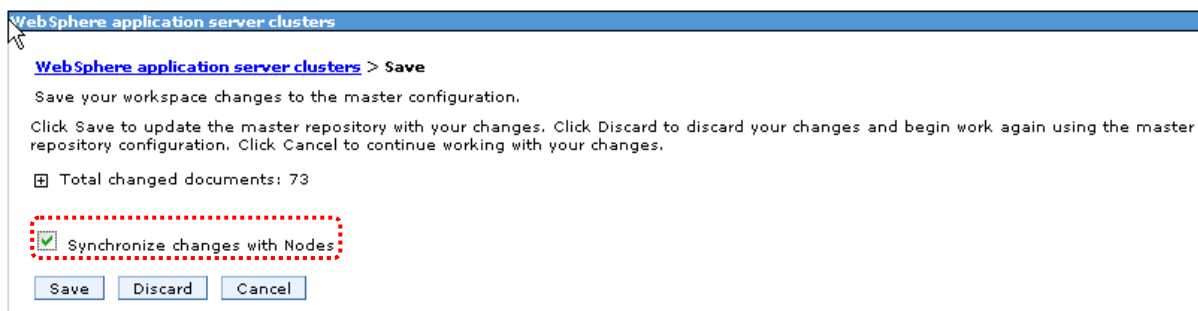


그 이유는 WAS ND 토폴로지 구조에 따른 Dmgr profile 만의 특성인데, Dmgr의 config 폴더가 실질적으로 Master repository 라 불리며 해당 Cell 의 WAS에 대한 모든 설정을 가지고 있는 1차 main repository 이기 때문입니다. 사용자가 WAS 관리콘솔을 통해서 설정을 변경하면, 이는 우선 Dmgr profile 의 master repository 에 저장됩니다. 그리고 나서, 각각의 profile, 즉 각 node 에서 필요한 해당 부분(변경된 부분)이 복사되어서 전송됩니다. 이렇게 복사된 설정 xml 파일들이 실제로 해당 node 의 WAS 들이 시작되면서 적용되는 것입니다. 따라서, 해당 node 들은 주기적으로 Dmgr 에 접속하여 변경된 사항이 있는지 확인하고 변경된 것이 있으면 해당 xml 파일을 복사해서 가지고 오게 됩니다. 이와 같은 작업을 동기화(synchronize) 작업이라고 합니다.



참조: WAS v6.1 redbook

당연하게도 동기화 작업은 필요한 경우에 수동으로 호출 할 수도 있습니다. 이전 WAS ND Clustering 강좌에서 설정을 변경하고 나서 저장하지 않고 보여드렸던 검토(review) 부분을 누를 경우에 나오는 **synchronize changes with nodes** 옵션이 변경된 설정을 수동으로 바로 동기화 하는 옵션입니다. (Dmgr 에 저장된 설정 xml 파일을 해당 node 로 복사하는 작업)



또한, 관리콘솔의 System administration > Nodes 메뉴에서 각 Node 에 대한 동기화 상태를 관리 콘솔에서 바로 확인 가능하며 수동으로 동기화도 가능합니다.

Nodes

Use this page to manage nodes in the application server environment. A node corresponds to a physical computer system with a distinct IP host address. The following table lists the managed and unmanaged nodes in this cell. The first node is the deployment manager. Add new nodes to the cell and to this list by clicking Add Node.

Preferences

Add Node Remove Node Force Delete **Synchronize** Full Resynchronize Stop

Select	Name	Host Name	Version	Discovery Protocol	Status
	T400B92090901CellManager01	T400B92090901.kr.ibm.com	ND 7.0.0.7	TCP	+
<input type="checkbox"/>	T400B92090901Node01	T400B92090901.kr.ibm.com	ND 7.0.0.7	TCP	+
<input type="checkbox"/>	T400B92090901Node02	T400B92090901.kr.ibm.com	ND 7.0.0.7	TCP	!

마지막으로 System administration > Node agents 메뉴에서 해당 노드 에이전트를 클릭한 후 file synchronization service 메뉴를 클릭하면 해당 node 에 대한 동기화 서비스와 주기를 하단처럼 사용자에게 적합하게 설정하실 수도 있습니다.

[Node agents](#) > [nodeagent](#) > **File synchronization service**

Use this page to configure the file synchronization service. 1 node agent. It ensures that configuration changes made to repositories.

Configuration

General Properties

☒ Enable service at server startup

Synchronization interval
37 minutes

☒ Automatic synchronization

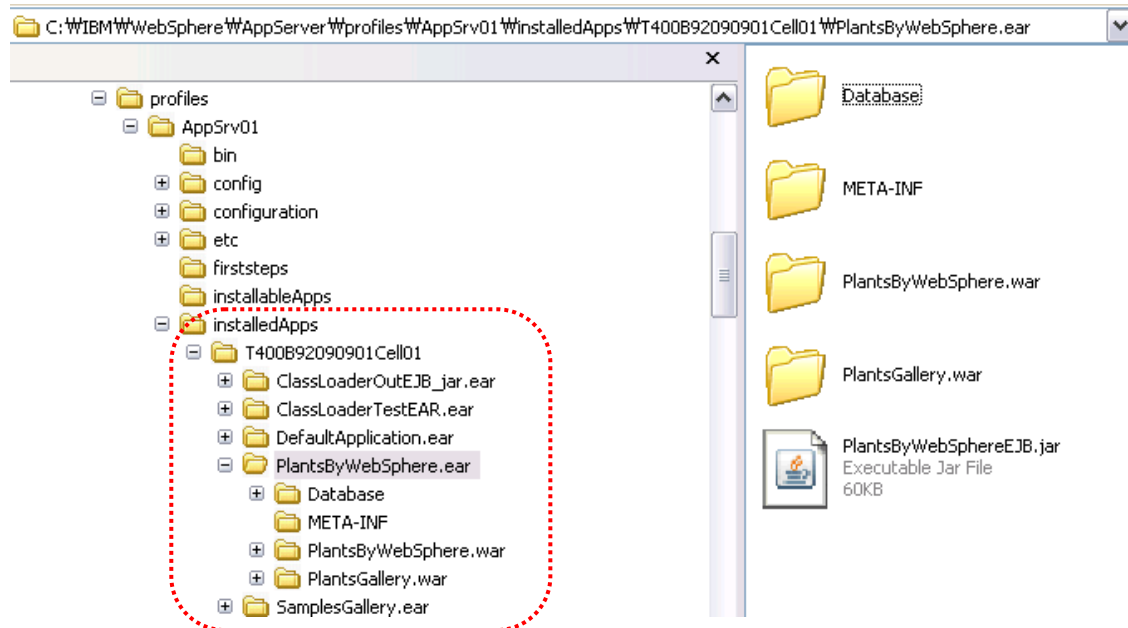
☐ Startup synchronization

이런 동기화 작업과 설정파일들은 IBM WAS ND 구조를 이해하는데 상당히 중요하기 때문에 꼭 숙지하시기 바랍니다.^^&

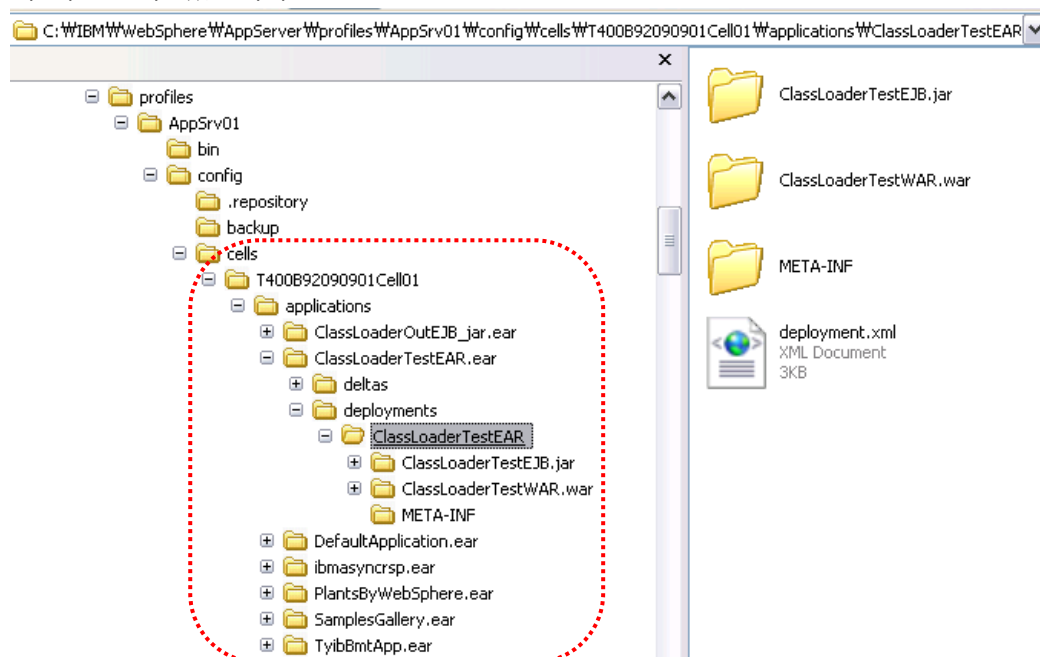
Part 3. Profile 안에 실제로 설치된 어플리케이션 설정

Part 2 에서는 profile 안의 config 를 통해서 실제 설정들이 저장되는 폴더의 위치와 방법등을 배웠습니다. 이번 파트에서는 맨 위에서 언급한 것 처럼 installedApps 에 저장되는 어플리케이션에 대한 추가 설명을 진행하도록 하겠습니다.

installedApps 폴더를 확인하시면 하단처럼 현재 사용하시는 Cell 이름의 밑에 각 어플리케이션이 설치된 것을 확인하실 수 있습니다. (단, 어플리케이션 설치시 폴더를 지정하지 않은 채 기본으로 설치된 경우에만 installedApps 폴더에 설치됩니다.)

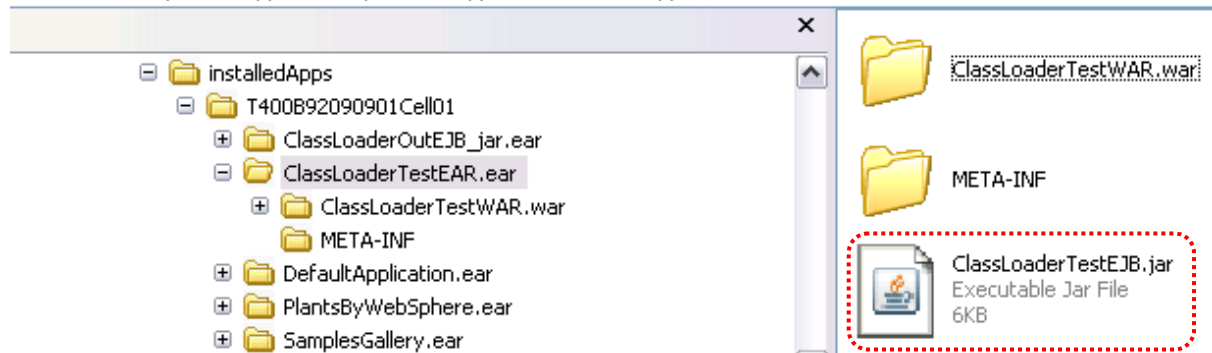


특pecially 어려운 것 없이 바로 보이긴 하지만, 사실 이게 전부가 아닙니다. 하단의 그림을 참조하시면 바로 이해하시겠지만 config 폴더 밑에 Cells > applications 밑에 동일한 어플리케이션을 하나 더 확인할 수 있습니다.



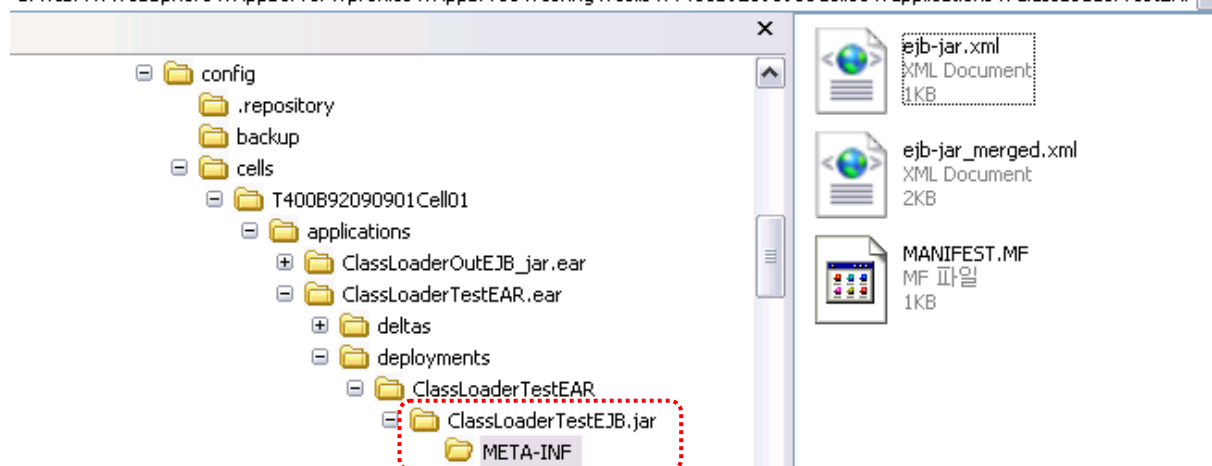
바로 이 부분이 많은 분들이 헷갈려 하시면서, 어려워 하는 부분입니다. WAS 관리콘솔을 이용해서 어플리케이션을 실제로 배포하면, 한 군데가 아니라 위와 같이 두 군데에 저장됩니다. 그러나 두 폴더를 좀 더 찬찬히 비교해보면 차이점을 알 수 있습니다. installedApps 폴더 밑에는 어플리케이션에 관련된 프로그램 소스 파일들과 설정 파일들을 하단처럼 확인할 수 있습니다.

C:\WIBM\WWebSphere\WAppServer\Wprofiles\WAppSrv01\WinstalledApps\WT400B92090901Cell01\WClassLoaderTestEAR.ear



그러나, config 폴더 밑의 어플리케이션은 실제 프로그램 소스 파일들은 존재하지 않은 채 폴더 구조와 ejb-jar.xml 같은 설정 파일들만 존재 합니다.

C:\WIBM\WWebSphere\WAppServer\Wprofiles\WAppSrv01\Wconfig\Wcells\WT400B92090901Cell01\Wapplications\WClassLoaderTestEAR



위와 같이 분리된 이유는 어플리케이션 설치시 실제 어플리케이션과 설정 파일들은 지정된 위치에 저장되지만 추가적으로 설정을 관리하는 config 밑에도 한벌 더 존재하게 되기 때문입니다. WAS 입장에서는 통합관리를 위하여 이렇게 두 폴더에 실제 돌아가는 어플리케이션 프로그램 소스와 어플리케이션 프로그램에 대한 설정을 분리하여 관리하는 것 입니다.

따라서, 여기서 보다 더 중요하게 기억할 것은 실제로 WAS 가 구동될 때 어플리케이션에 대한 설정 부분은 실제 설치가 된 installedApps 폴더 밑에 있는 것이 아니라 config 밑에 있는 어플리케이션의 설정 정보를 참조해서 구동된다는 점입니다. 그렇기 때문에 manually 어플리케이션의 설정이나 dd 파일, web.xml 파일을 직접 수정하려고 한다면 installedApps 밑에 있는 것이 아니라 config > cells > applications 폴더 밑에 있는 어플리케이션의 설정이나 dd 파일, web.xml 파일을 수정해야지만 변경된 것을 확인할 수 있습니다. (그렇지 않으면 설정을 직접 변경을 무한 반복해도 동일한 화면만 계속 보게 되는 실수를 하게 되며 사실, 이 부분이 많은 분들이 실수하시는 부

분입니다.)

이 부분이 다른 WAS 의 구조와는 달라서 많이 헷갈려 하시지만 통합 관리와 안정적인 관리를 지향하는 WebSphere 만의 장점으로 생각해 주시면 고맙겠습니다..^^&

단, 위와 같은 구조가 너무 불편하다고 느껴지고 운영 환경의 필요에 의해서 반드시 설치된 어플리케이션에서 설정 파일까지 같이 참조할 수 있도록 해야 한다고 결정하셨다면 관리콘솔에서 Enterprise applications > 해당 어플리케이션 > Application binaries 메뉴를 클릭하여 Use configuration information in binary 라는 옵션을 체크하시면 됩니다. 해당 옵션을 사용하면 installedApps 에 설치된 어플리케이션 파일 및 설정 파일을 모두 installedApps 에 설치된 것으로 참조합니다. 즉, config 밑의 applications 폴더에 있는 web.xml 같은 설정을 사용하지 않는 옵션입니다.

[Enterprise Applications](#) > [DefaultApplication](#) > **Application binaries**

Use this page to configure the location and distribution of application binary files.

Configuration

General Properties

* Location (full path)

`${APP_INSTALL_ROOT}/${CELL}`

☐ Use configuration information in binary

☒ Enable binary distribution, expansion and cleanup post uninstallation

참고: Enable binary distribution, expansion and cleanup post uninstallation 옵션은 해당 어플리케이션이 맵핑된 서버의 node 마다 어플리케이션을 자동으로 배포하고 설치하는 옵션입니다. 따라서 해당 옵션을 사용하지 않으면(기본적으로 사용으로 설정) WAS ND 의 Clustering 환경을 구축했다고 하더라도 node 마다 어플리케이션을 복제해서 사용하는 것이 아니라 설치 위치에 미리 어플리케이션을 설치해야 합니다. 즉, 어플리케이션을 사용만 할뿐 배포하지 않으며 설치된 하나의 어플리케이션 binary 를 각 node 들이 공유해서 사용합니다.

(단, 어플리케이션의 web.xml 같은 설정들은 매핑되어진 Server 의 node 의 config 밑의 폴더에 복제되어집니다.)

여기까지 잘 이해가 되시나요? WebSphere 의 경우에는 다른 WAS 와는 조금은 다른 구조를 가지고 있고 이해하기가 조금 어려울 수도 있지만 이전에 말씀드린 것처럼 통합 관리와 안정적인 관리를 위하여 만들어진, 견고한 구조를 위하여 사용하고 있는 것입니다. 따라서 다시 한 번 위에서부터 천천히 WAS 가 가지고 있는 폴더와 설정 구조를 곰곰이 생각해 보고 이해하시기 바라겠습니다.

니다. 이번 강좌는 특정 기능이나 기술에 대한 설명 대신에 WAS 관리자라면 알아두어야 할 WAS의 폴더 구조와 설정 파일들 등에 대한 설명을 하였습니다. 그럼 이번 강좌는 여기서 마무리 하도록 하겠습니다. 이만~~~~ ^^&

참고 1) IBM WebSphere Application Server v7.0 InfoCenter

http://publib.boulder.ibm.com/infocenter/wasinfo/v7r0/index.jsp?topic=/com.ibm.websphere.nd.multipiplatform.doc/info/welcome_nd.html

참고 2) IBM WebSphere Application Server v7.0 InfoCenter

- File synchronization service settings

http://publib.boulder.ibm.com/infocenter/wasinfo/v7r0/topic/com.ibm.websphere.nd.doc/info/ae/ae/uagt_rsynchronservice.html?resultof=%22%73%79%6e%63%68%72%6f%6e%69%7a%65%22%20%22%73%79%6e%63%68%72%6f%6e%22%20

※이 자료의 저작권은 작성자에게 있으며 유포는 자유로이 허용되나 상업적으로 이용은 금합니다.