Setup

1. Jenkins 홈페이지에서 Window 버젼 다운로드
2. 설치가 완료되면 자동으로 localhost:8080 으로 접속됨
3. 이 때 8080 port가 사용 중이면 404에러 발생
   * Jenkins가 설치된 경로로 이동(C:\Program Files (x86)\Jenkins)해서 Jenkins.xml 파일 수정 (port를 8080이 아닌 다른 값으로 설정)
4. 웹 페이지가 출력되면 기본 설치는 완료.

Git 연동

1. Jenkins 관리 -> 플러그인 관리 -> Git 관련 플러그인 설치 (Github을 이용하면 Github 관련 Plugin을, Bitbucket을 이용하면 Bitbucket 관련 Plugin을 설치)
   * Git Plugin
   * Github Plugin
   * Bitbucket Oauth Plugin
   * Bitbucket Plugin
2. 새로운 Item 선택
   * Free-stype로 생성
3. Git plugin을 사용하려면 몇가지 환경 설정이 필요하다.
   * 참고 : http://computercamp-cdwilson-us.tumblr.com/post/48589650930/jenkins-git-clone-via-ssh-on-windows-7-x64
   * Jenkins 관리 -> 시스템 구성 -> Git -> Path to Git executable
   * git.exe 연결이 제대로 되어 있지 않아서 빨간 글씨로 오류가 나 있을 것이다.
   * Git이 설치된 경로(C:\Program Files\Git) 에서 bin/git.exe 경로를 입력한다.
   * C:\Program Files\Git\bin\git.exe
   * 적용 버튼 클릭
4. 초기의 Jenkins를 구동시키는 것은 Local System account이지 user 계정이 아니다. Local System account는 SSH key 가지고 있지 않거나 나 known\_hosts이 설정되어 있지 않다. 그러므로 git clone을 수행하면 clone에 실패할 것이다.
   * 이를 해결하기 위해서는 ssh key를 gen 해서 git에 add 해줘야 한다.
     + http://knight76.tistory.com/entry/git-Permission-denied-%EB%AC%B8%EC%A0%9C-%ED%95%B4%EA%B2%B0
     + https://help.github.com/articles/error-permission-denied-publickey/
     + http://uiandwe.tistory.com/992
     + https://backlogtool.com/git-guide/kr/reference/trouble-shooting.html
   * 내 경우에는 설정에 Git 계정과 비밀번호를 등록했더니 해결되었다.
     + Jenkins 관리 -> 시스템 설정 -> Git Plugin에 username과 email 입력
     + Jenkins 홈 -> Git 프로젝트 item을 선택 -> 구성 -> 소스코드 관리 -> Git -> Credentials에 계정 Add

MSBuild 설정

1. Visual Studio 프로젝트를 빌드하기 위해 MSbuild를 사용
2. Jenkins 관리 -> 플러그인 관리 -> MSBuild Plugin 설치 -> Jenkins 재시작
3. Jenkins 관리 -> 시스템 설정 -> MSBuild -> Add MSBuild
   * Name : .Net Framework 4.0 (이름은 알아서 지정)
   * Path to MSBuild : C:\Windows\Microsoft.NET\Framework\v4.0.30319\MSBuild.exe (환경에 맞는 MSBuild를 선택)
4. Jenkins 홈 -> Git 프로젝트 item을 선택 -> 구성 -> Build -> Add build step -> Build a visual studio project or solution using MSBuild 선택
   * MSBuild Version : 3번에서 지정한 Name 선택
   * MSBuild Build File : 빌드할 솔루션의 경로지정
5. Jenkins 홈 -> 빌드 시작
6. Console Output 을 확인하면 빌드 진행 상황을 체크 할 수 있다.
7. 빌드시 프로젝트 설정대로 빌드를 하지 않아 오류가 발생했다.
   * MSBuild 수행할때 옵션이 필요
     + http://acidpop.tistory.com/118
   * 아직 정확한 문제파악이 되지 않아 알아보는 중
     + msbuild 관련 슬라이드쉐어 : http://www.slideshare.net/kaistizen/ss-10084687
     + 빌드 순서에 대한 문제 해결: http://resisa.tistory.com/104
   * MSBuild 개념 잡기
     + http://ccambo.blogspot.kr/2014/02/msbuild-1.html
     + http://megustaria.tistory.com/3
   * 알게된 정보
     + msbuild는 c++ 빌드하기가 까다롭다?
       - vcbuild
       - devenv 고려
     + visual studio 2005 프로젝트를 msbuild로 빌드하려면 .Net framework 2.0 버젼으로 해본다
     + 현재 프로젝트의 .net framework 버젼을 확인한 후 해당 버젼으로 시도
     + vs2005에서 msbuild 사용하기 : http://stackoverflow.com/questions/832602/msbuild-with-visual-studio-2005
     + DependsOnTargets 속성으로 빌드 순서를 정할 수 있다

Devenv 사용

* MSBuild를 사용해서 MStar 프로젝트를 빌드해보니 Dependency 때문에 빌드가 제대로 되지 않았다. (MSBuild는 프로젝트간 의존성을 무시하고 순서대로 빌드) 그래서 devenv를 사용하는 방법으로 알아보았다.
* 빌드 커맨드 : devenv.exe [솔루션 경로] [option]
* Example
  + “C:\Program Files (x86)\Microsoft Visual Studio 8\Common7\IDE\devenv.com” /INCREMENTAL:NO /useenv “C:\work\jenkins\_test\jenkins\_test.sln” /Build “Debug|Win32” /out “C:\buildlog.txt” (/Build를 수행할 때 Debug 환경에 Win32 플랫폼으로 빌드하고 출력되는 로그는 C:\buildlog.txt에 저장)
  + devenv.exe와 devenv.com으로 명령 수행이 가능한데 devenv.exe는 콘솔 출력이 없고 devenv.com은 콘솔에 출력을 해준다.
  + /out 옵션을 사용하면 인자로 전달되는 경로에 로그가 기록된다.
* 참고
  + https://msdn.microsoft.com/ko-kr/library/b20w810z.aspx

Jenkins에서 batch command 관리자 권한으로 실행

* devenv를 이용해서 빌드 시 관리자 권한으로 수행해야 정상적으로 빌드가 되는데 이를 커맨드 상에서 지정하는 방법이 기본적으로는 없고, Jenkins 서비스 자체를 관리자 권한으로 시작하는 방법을 사용해야 한다.
  + 서비스 -> Jenkins 에서 속성으로 진입 -> 로그온 -> 계정 지정 -> 관리자 지정
  + Jenkins 서비스 재시작
* 참고
  + http://stackoverflow.com/questions/27413261/run-batch-file-as-administrator-on-jenkins

빌드 시 매개변수 설정

1. Jenkins Item의 구성에 들어가서 “이 빌드는 매개변수가 있습니다.”에 체크한다.
2. 상황에 맞는 매개변수를 선택한 후 값을 지정한다.
3. build 명령에서 매개변수 이름 앞뒤에 %를 붙이면 빌드 시 해당 매개변수의 값으로 치환된다.
   * ex) 매개변수명 : Command -> %Command%

이슈 해결

1. Windows Batch command를 이용해서 프로그램을 실행 시킬 때 코드 내에 “Environment.CurrentDirectory”를 이용하거나 “Directory.GetCurrentDirectory()”를 사용해서 경로를 가지고 오는 경우 프로젝트의 작업 디렉토리가 아닌 파일을 실행시킨 위치를 가져오므로 잘못된 위치를 가리킬 수 있다. 프로젝트의 작업 디렉토리를 가져오기 위해서는 “AppDomain.CurrentDomain.BaseDirectory”를 사용한다.
2. 젠킨스를 이용해서 다른 피씨에 배포를 하는 경우 경로를 찾지 못하는 문제가 발생했다. 로컬에서는 정상 동작하는데 외부 피씨로 배포하는 경우에만 자꾸 경로를 찾지 못해서 한참을 헤매던 중 젠킨스로 실행 될 때 실행하는 사용자 계정을 찍어보니 AUTHORITY\SYSTEM 이었다. 이는 젠킨스 서비스의 설정이 로컬 시스템 계정으로 되어있기 때문이었는데 원격 피씨의 공유 폴더에 이 계정에 대한 권한이 없어서 였다. 그래서 공유폴더에 권한을 가지고 있는 계정으로 지정해서 서비스를 재시작하니 해결되었다.

출처: <https://yongho1037.tistory.com/601> [개발 노트]