

SHARE
NDC11

NEXON
DEVELOPERS
CONFERENCE
2011

레가시 프로젝트의 빌드 자동화



라이브개발 2본부 5실 컴뱃암즈 팀
최재훈

발표자 소개

최재훈 [AndromedaRabbit](#)

2004년 슈어엠 – SMS / MMS

2007년 잠깐 복학

2007년 SK 아이미디어 – 미니라이프

2009년 오픈소스 - 얼그레이

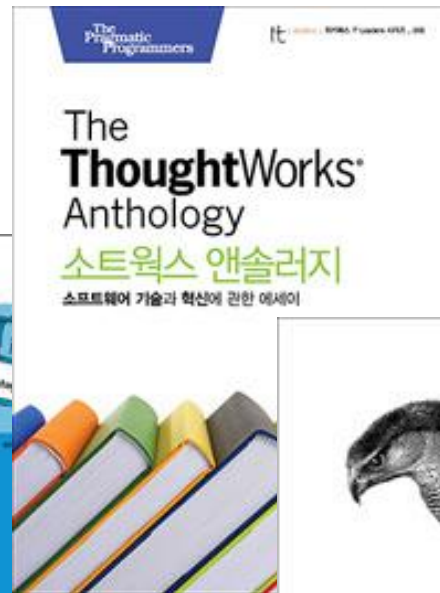
2009년 EA Korea – 피파 온라인 2

2010년 잠깐 방황

2010년 넥슨 – 컴뱃암즈

발표자의 곁가지 활동

<http://andromedarabbit.net/wp/>



여러분이 얻어 갈 것

자동화가 어렵다는 편견을 버린다

자동화가 쉽다는 편견도 버린다

대규모 레가시 프로젝트도 된다는 용기를 얻는다

게으른 삶을 되찾는다

사례

컴뱃암즈

약 8개월, 레가시, 클라이언트/서버, 주요 분석 사례

얼그레이

약 2년, 신규, 엔진/빌드 도구, 예제

그 밖의 소소한 경험담



2008 © Nexon Corporation and Nexon America Inc. All Rights Reserved.



관히 자랑질



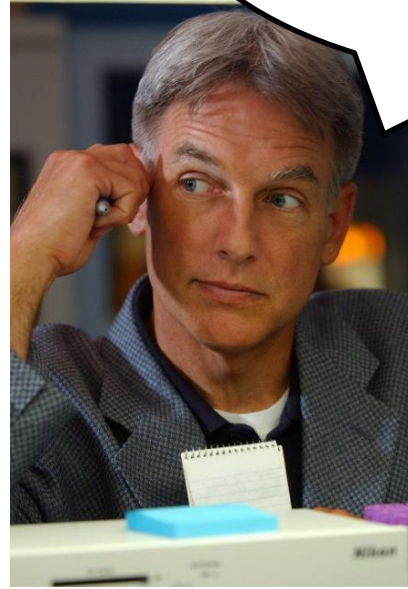
어느 날,

그게 뭔가요?
먹는 건가요?

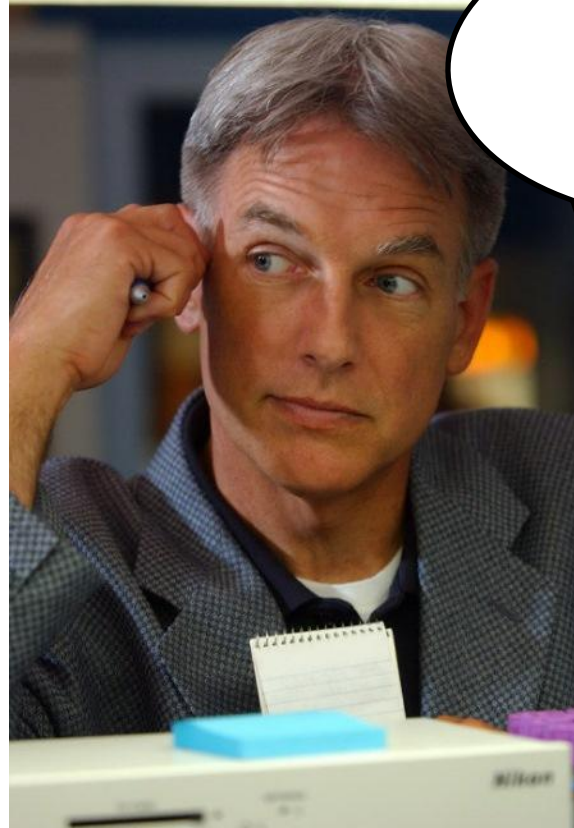


빌드 자동화라고
들어봤나?

그거 하면
좋다던데?



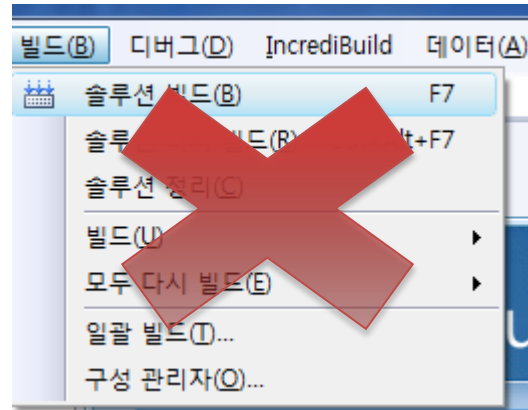
날벼락이!



네가
해보지?

나더러 어쩌라고

우선 내가 할 줄 아는 걸 정리해 보자



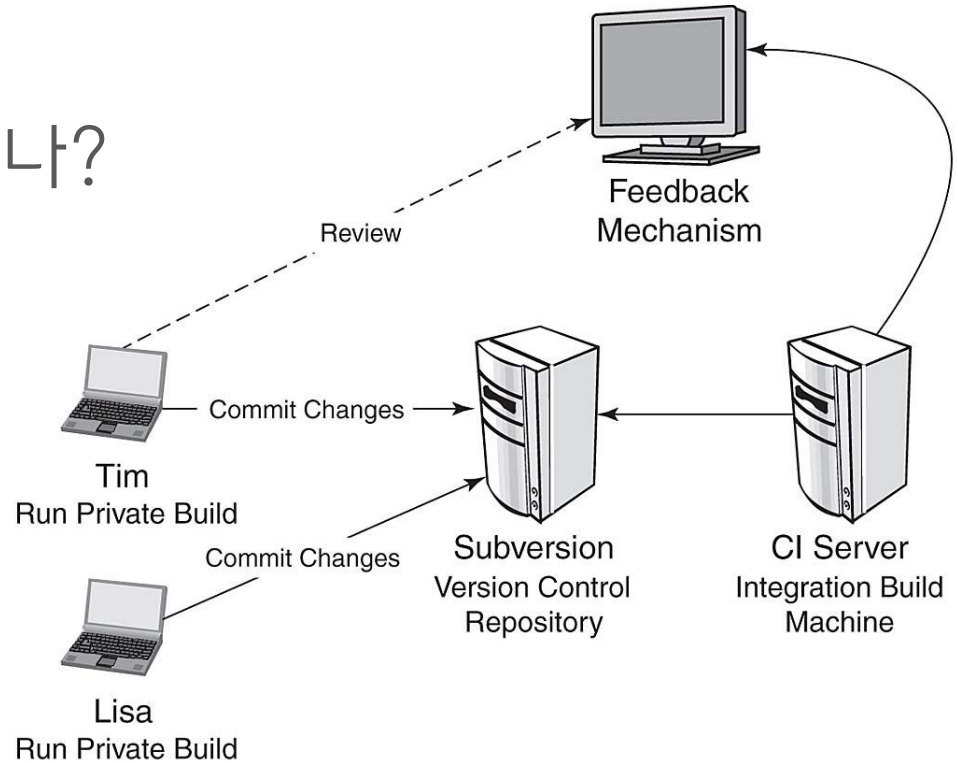
F7은 빌드 자동화가 아니다.

빌드 자동화란?

책에 따르면

소스 서버는 있으니,
CI 서버만 있으면 되나?

CI
= Continuous Integration
= 지속적인 통합



CI 서버란?

CruiseControl .NET 의 경우,

The screenshot displays the CruiseControl.NET server logs and the CCTray interface. The logs show the server version (1.3.0.2918) and runtime information. The CCTray interface shows a tree view of build queues and a table of build activities. A yellow tooltip with a red 'X' icon indicates a 'Broken build' and states 'Recent checkins have broken the build.' The table below shows the current state of various build projects.

Project	Server	Activity	Detail	Last Build Label	Last Build Time
CommonLibrary	buildfarm-01.minispace.net	Sleeping	Next build check: 오전 11:47:44	12	2008-03-14 오전 11:47:44
Cuckoo-MainBuild	buildfarm-01.minispace.net	Pending		189	2008-03-17 오전 11:47:44
Cuckoo-Release	buildfarm-01.minispace.net	Building		161	2008-03-17 오전 11:47:44
Cuckoo-SubBuild	buildfarm-01.minispace.net	Pending		267	2008-03-17 오전 11:47:44
Cuckoo-x64-Release	buildfarm-01.minispace.net	Pending		139	2008-03-17 오전 11:47:44
CommonLibrary-VS2008	buildfarm-02.minispace.net	Sleeping	Next build check: 오전 11:46:21	6	2008-03-14 오전 11:46:21
Cuckoo-Mainbuild-VS2008	buildfarm-02.minispace.net	Pending		32	2008-03-17 오전 11:46:21
Cuckoo-SubBuild-VS2008	buildfarm-02.minispace.net	Pending		26	2008-03-17 오전 11:46:21
Cuckoo-Release-VS2008	buildfarm-02.minispace.net	Pending		19	2008-03-17 오전 11:46:21
Cuckoo-x64-Release-VS2008	buildfarm-02.minispace.net	Building	14 minutes estimated remaining	7	2008-03-17 오전 11:46:21

C# 서버를 도입했으니 끝?

Visual Studio만 자동으로 돌리면 끝 아닌감?

이라고 대부분의 팀이 만족하고 만다.



```
<devenv>  
  <solutionfile>Earlgrey.BuildTools.sln</solutionfile>  
  <configuration>Debug</configuration>  
  <executable>c:\program files\Microsoft Visual Studio  
    .NET\Common7\IDE\devenv.com</executable>  
  <buildTimeoutSeconds>10</buildTimeoutSeconds>  
</devenv>
```

게임 하나 뽑으려면

.exe, .dll 만 있다고 되나?

쌍팔년도도 아닌데 그래픽 없는 게임 있나?

데이터베이스! 온라인 게임의 저주!

핵실드! 망할 해커들!

인스톨러. 예쁘게 포장해야 팔리지

다 만들었으면,

퍼블리셔에게 던져줘야지

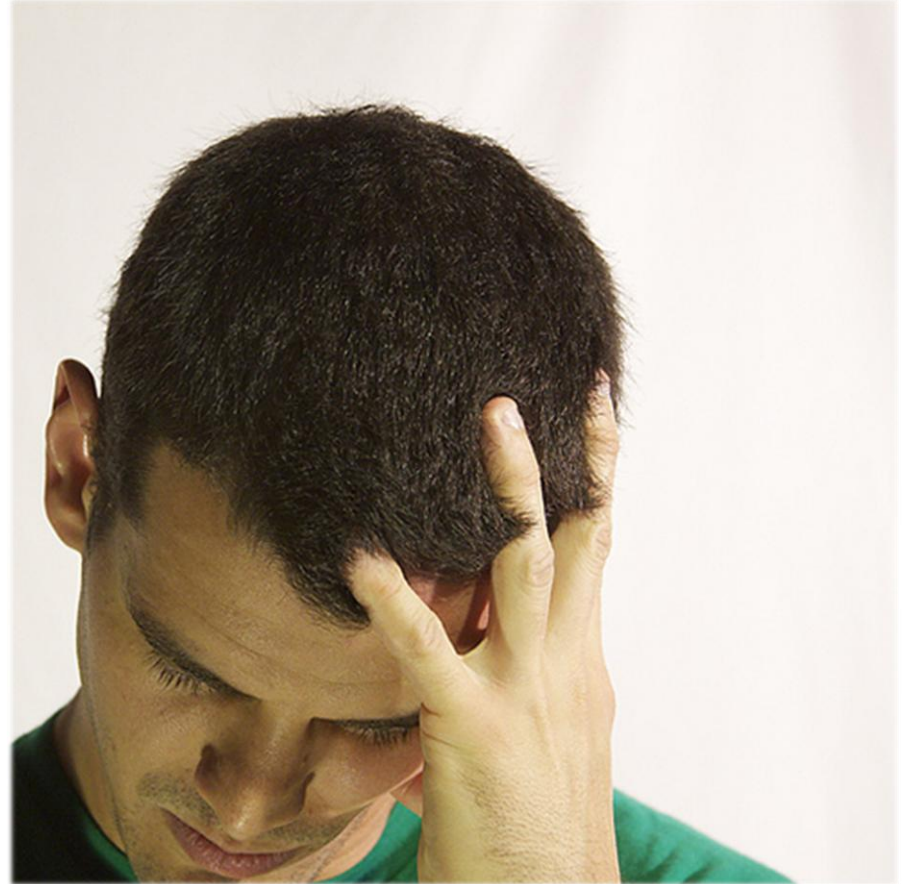
백업은 안 하나?



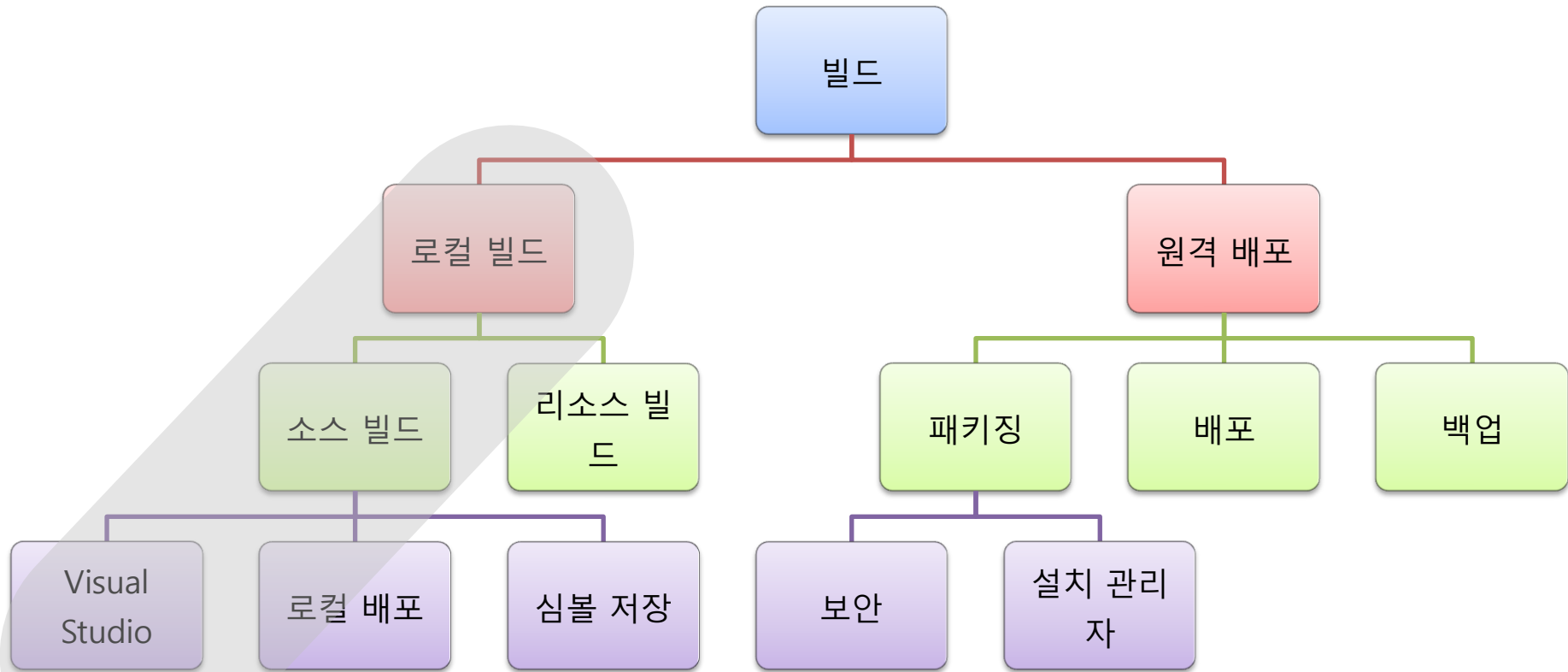
설상가상

기술 빛

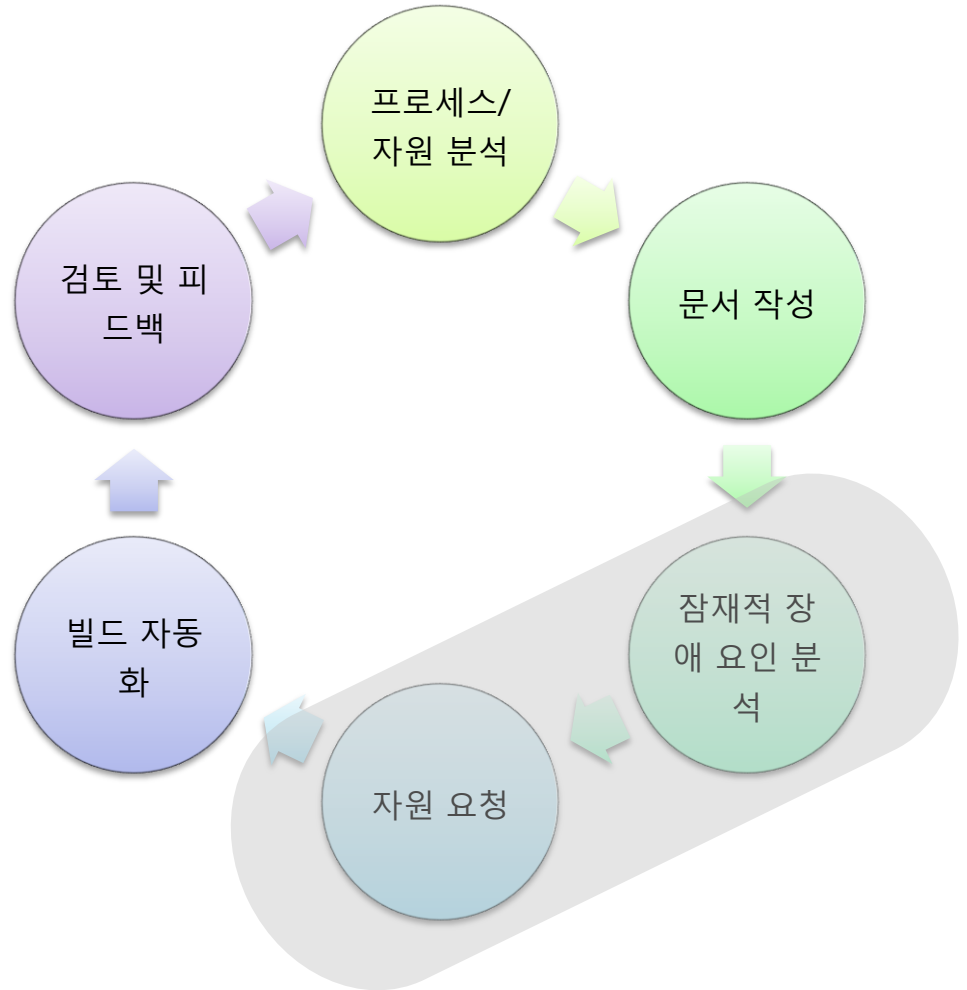
5년짜리 빛이라...



그래서 제대로 만들면,



당신도 할 수 있... 을 걸?



이대로 해 봅시다.

컴뱃암즈의 사례를 따라,



기존 프로세스를 분석해보니

제일 큰 문제는 속도!

A4용지 24쪽짜리 수작업 빌드 과정

본 발표자료와 달리 그림 따위 없는 불친절한 문서임



기존 프로세스를 분석해보니

다양한 환경

플랫폼 (WIN32)

구성 (DEBUG, RELEASE, FINAL)

지역 (한국, 북미, 유럽, 남미)

배포 (개발자, 내부 테스트, 출시)

$1 \times 3 \times 4 \times 3 = 36$ 가지



기존 프로세스를 분석해보니

분산된 자원

빌드 머신에만 있는 패키징 도구
따로 관리되는 서버/클라이언트
→ 버전 관리 저장소에 모두 넣었다

버전 관리 안 하는 데이터베이스

별도로 설치해야 하는 라이브러리

기존 프로세스를 분석해보니

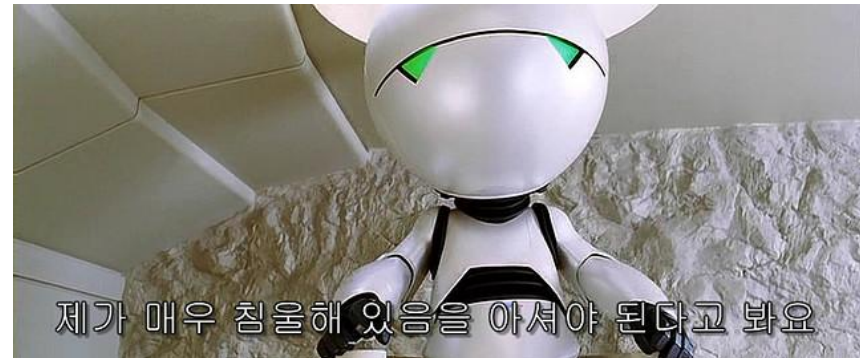
자동화가 힘든 구성 요소

명령 줄을 지원하지 않는 도구

패키징 도구, Torrent

명령 줄 지원이 미약한 도구

AlienBrain



기존 프로세스를 분석해보니

지역별로 다른 배포 방식

웹 하드는 자동화가 어려움



그래서,

TO BE CONTINUED

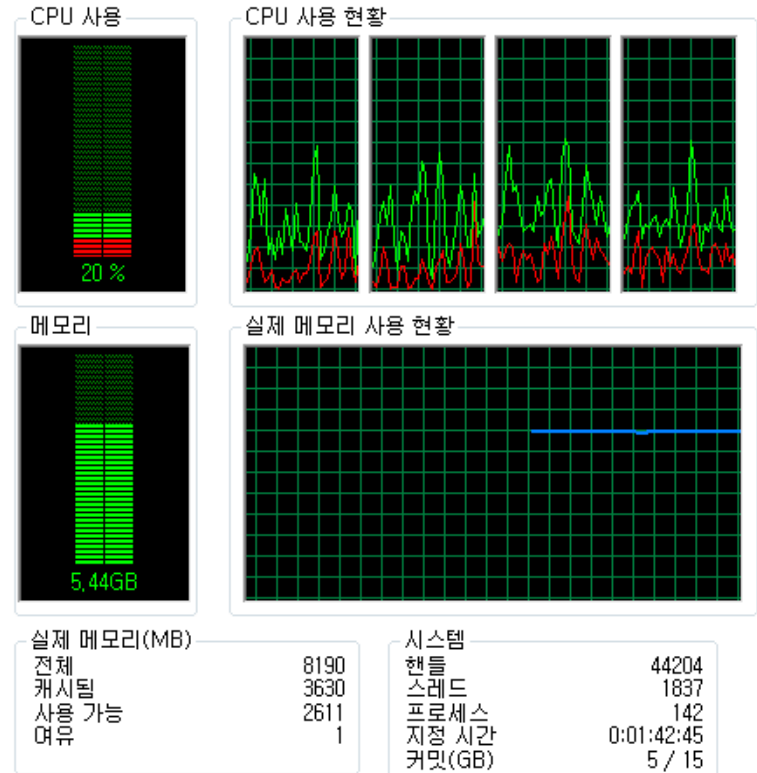
빌드가 2시간씩 걸려서야

하드웨어

빌드 머신
패치 머신
파일/백업 서버
소스/웹 서버 머신

공통

고성능 디스크
SSD, 하이브리드 HDD, RAID
대용량 RAM
8GB, Windows x64



- I/O 작업
- 메모리 고갈
- 페이지 스왑
- 성능 저하

성능 개선

결론부터?

무려 **4배!!!**

2시간에서 30분으로



성능 개선의 요인

하드웨어 업그레이드의 결과

똑같은 빌드 스크립트를 여러 머신에서 돌려 보니

약 **35%** 향상

	예전 빌드 머신	예전 예비용 빌드 머신	로컬 머신	새 빌드 머신
SVN 관련 작업	127,734	6,734	13,391	7,025
파일 복사/삭제	252,732	186,448	76,085	41,563
리소스 컴파일	340,107	319,056	134,217	93,240
컴파일	906,854	1,086,963	785,376	666,004
핵심드 등	174,015	32,203	162,817	172,939
패키징	1,017,259	1,287,553	1,088,341	802,845
기타	2,203	2,952	1,992	1,413
총합 (분)	47	49	38	30

두 번째 테스트 (일반적인 패키징)

	예전 빌드 머신	예전 예비용 빌드 머신	로컬 머신	새 빌드 머신
SVN 관련 작업	126,812	83,029	65,740	12,064
파일 복사/삭제	165,919	187,683	77,674	38,655
리소스 컴파일	357,639	328,509	97,560	70,578
컴파일	858,886	1,065,792	760,654	681,558
핵심드 등	196,047	272,698	160,521	143,849
패키징	967,041	1,344,115	1,052,283	806,583
기타	2,843	2,859	2,329	1,447
총합 (분)	45	55	37	29

성능 개선의 요인

그럼 나머지는?

소프트웨어와 로직!

RoboCopy

병렬 파일 복사, 버퍼링, 바뀐 파일만 복사

SVN Diff

바뀐 리소스 파일만 빌드

IncrediBuild

분산 빌드, 9분 45초 → 5분 8초

괜한 고민 말자!



보스가 승인해 줄까?



어지간하면 해준다.

안 되면? 그럼 말구.

기존 빌드 과정의 재구성

주요 도구

명령줄 배치 스크립트

MSBuild (MS 사가 개발한 빌드 스크립트)

MSBuild Community Tasks (오픈 소스 확장기능)

RoboCopy (MS 사가 개발한 파일 복사 유틸리티)

Visual Studio

기존 빌드 과정의 재구성

원칙

통합 관리

모든 것은 소스 서버에
하나의 저장소
이식성

추적 가능

세밀한 오류 보고
.Bat의 최소화
전문적인 빌드 스크립트의 도입

기존 빌드 과정의 재구성

미리 만든 틀에 기존 과정을 끼워 넣는다

멍청한 기계도 이해 가능하게!

순차적으로!

```
<!-- Build / Rebuild / Clean -->
```

```
<Target Name="Build" DependsOnTargets="PrepareBuildTools; BuildSourceCodes;  
BuildResources; Build-$(DeploymentLevel)">  
  <Message Text="타겟: Build" />  
</Target>
```

```
<Target Name="Rebuild" DependsOnTargets="Clean; Build;  
Rebuild-$(DeploymentLevel)">  
  <Message Text="타겟: Rebuild" />  
</Target>
```

```
<Target Name="Clean" DependsOnTargets="Clean-$(DeploymentLevel)">  
  <Message Text="타겟: Clean" />  
</Target>
```






기존 빌드 과정의 재구성

배포 상황에 따라 따른 처리

개발자 dev

팀 내부 테스트 test

출시 final

 msbuild-localbuild.xml	2011-04-28 오후...	XML 문서
 msbuild-localbuild-common.xml	2011-05-02 오후...	XML 문서
 msbuild-localbuild-dev.xml	2011-04-28 오후...	XML 문서
 msbuild-localbuild-final.xml	2011-04-28 오후...	XML 문서
 msbuild-localbuild-test.xml	2011-04-28 오후...	XML 문서

```
<!-- msbuild-localbuild.xml -->
```

```
<Import Project="msbuild-localbuild-$(DeploymentLevel).xml"/>
```

소스코드 빌드

Visual Studio에게 맡기면 간단!

```
<!-- Source codes -->
<ItemGroup>
  <ProjectReferences Include=
    "$(SourceCodesDir)\CombatArms\CombatArms_All.sln">
    <Configuration>$(Configuration)_$(ShortLocalCode)</Configuration>
    <Platform>$(Platform)</Platform>
  </ProjectReferences>
</ItemGroup>

<Target Name="CompileSourceCodes" DependsOnTargets=
  "CompileSourceCodes-$(DeploymentLevel)">
  <Exec
    Command="$(Quot) $(DevEnv) $(Quot)
      $(Quot) % (ProjectReferences.FullPath) $(Quot) /$(BuildCondition)
      $(Quot) % (ProjectReferences.Configuration) |% (ProjectReferences.Platform)
      $(Quot) "
```

```
CompileSourceCodes:
```

```
· 빌드 조건: 'Rebuild'
```

```
· "devenv.com" "f:\Workspace\src\CombatArms\CombatArms_All.sln" /Rebuild
```

```
· "Final us|Win32"
```

```
</target>
```

통합 솔루션 파일

만약 솔루션 구성이 아래와 같다면

App1.sln = Shared.vcproj + App1.vcproj

App2.sln = Shared.vcproj + App2.vcproj

App3.sln = Shared.vcproj + App3.vcproj

```
<ItemGroup>
  <ProjectReferences Include="App1.sln" />
  <ProjectReferences Include="App2.sln" />
  <ProjectReferences Include="App3.sln" />
</ItemGroup>

<Target Name="Rebuild">
  <Exec
    Command="devenv.com
      $quot;% (ProjectReferences.FullPath) %quot;
      /Rebuild &quot;Release&quot;;"
  />
</Target>
```

통합 솔루션 파일

몇 번 빌드하게 되나?

Shared.vcproj : 3번

App1.vcproj : 1 번

App1.vcproj : 1 번

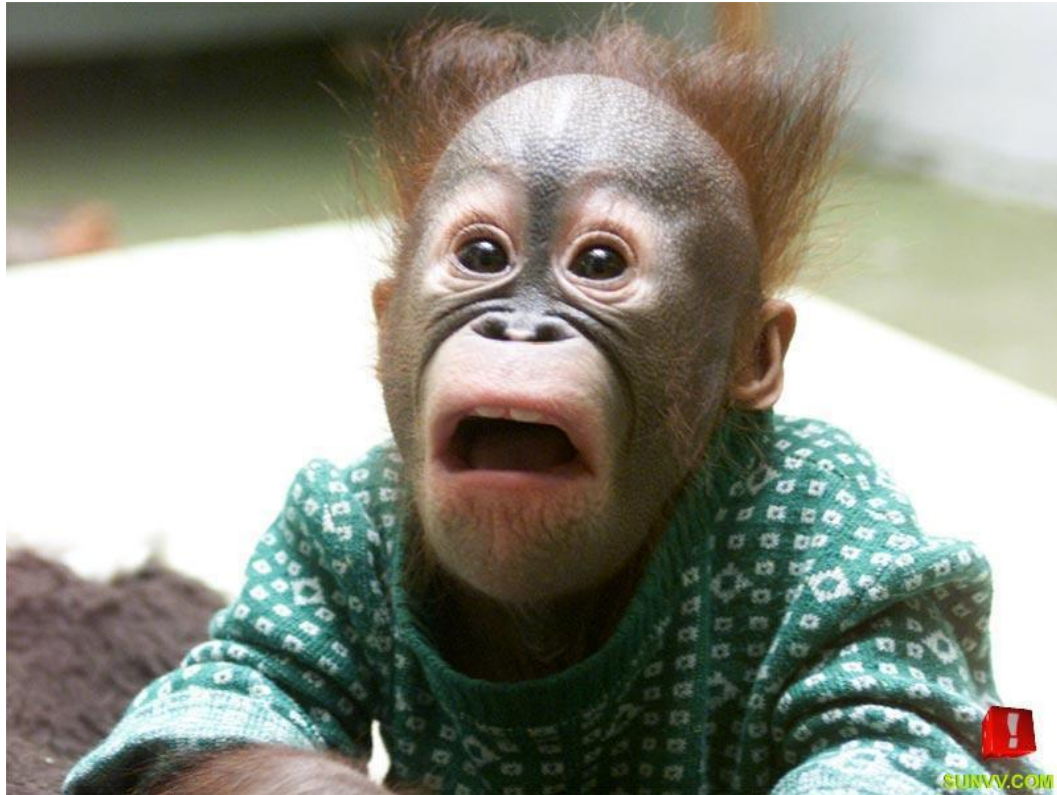
App1.vcproj : 1 번



4회면 충분할 것을 6회나!!!

VS를 안다고 자신하는가?

정말? 알면 됐구...



경로 지정은 이렇게 한다

절대 법칙

절대 경로 No!

하드코딩 No!

\$(SolutionDir) Avoid If Possible

공백 경로 고려 안 하기 No!

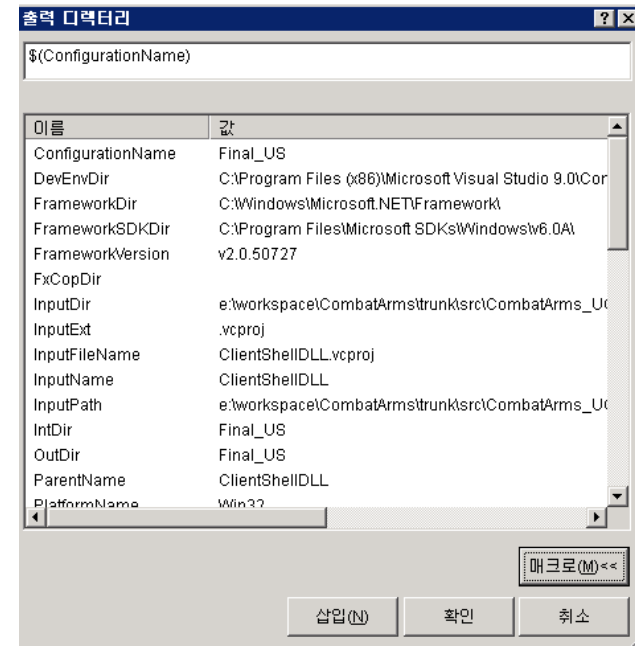
원칙: 통합 관리 이식성

추가 종속성

"\$(OutDir)#GmailClient.lib"

추가 포함 디렉터리

inc;"..#..#vendor#gtest-1.3.0#include"

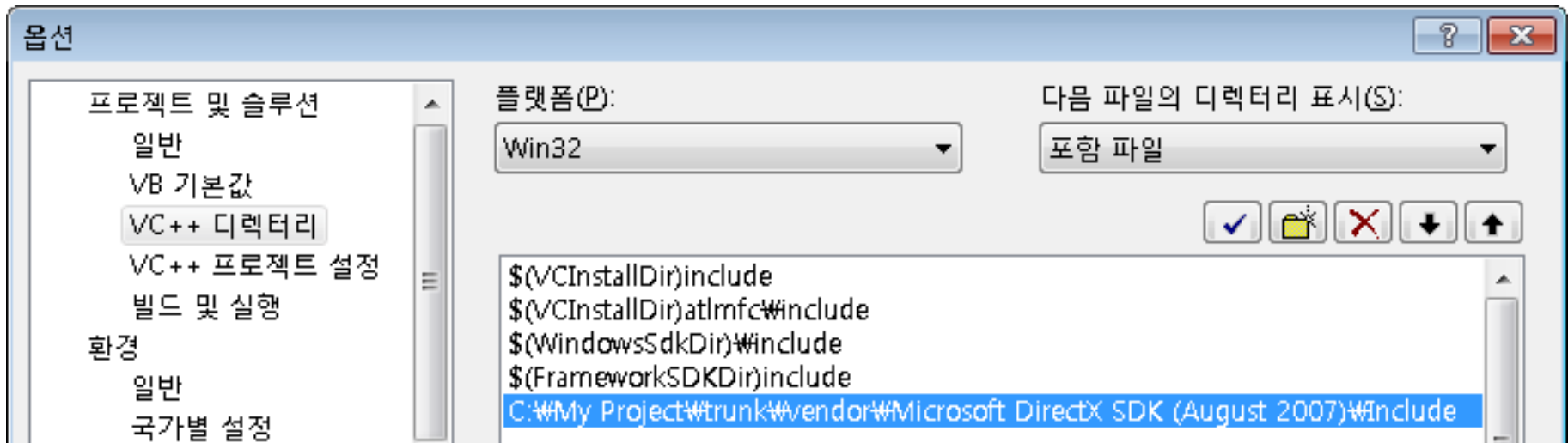


"\$(ProjectDir)..\My Dir\My Project.sln"

흔한 실수

라이브러리 참조

Visual Studio 의 환경설정 기능을 쓰지 말 것!
IncrediBuild 에서 문제를 일으키기도 한다.



빌드 이벤트 대신 출력 폴더

출력물 (.dll/.exe/.pdb)을 특정 폴더에 복사

The screenshot shows the 'Build Events' window in Visual Studio. The 'Command Line' tab is active, displaying the following command:

```
xcopy /y "$(TargetDir)$(TargetName).pdb" "..\..\PDB$(ConfigurationName)"
```

A smaller window titled '명령줄' (Command Line) is overlaid on top, showing the same command with expanded variables:

```
xcopy /y "$(TargetDir)$(TargetName).pdb" "..\Bin$(ConfigurationName)\"  
xcopy /y "$(TargetDir)$(TargetName).map" "..\Bin$(ConfigurationName)\"  
xcopy /y "$(TargetPath)" "..\Bin$(ConfigurationName)\"
```

Better Way

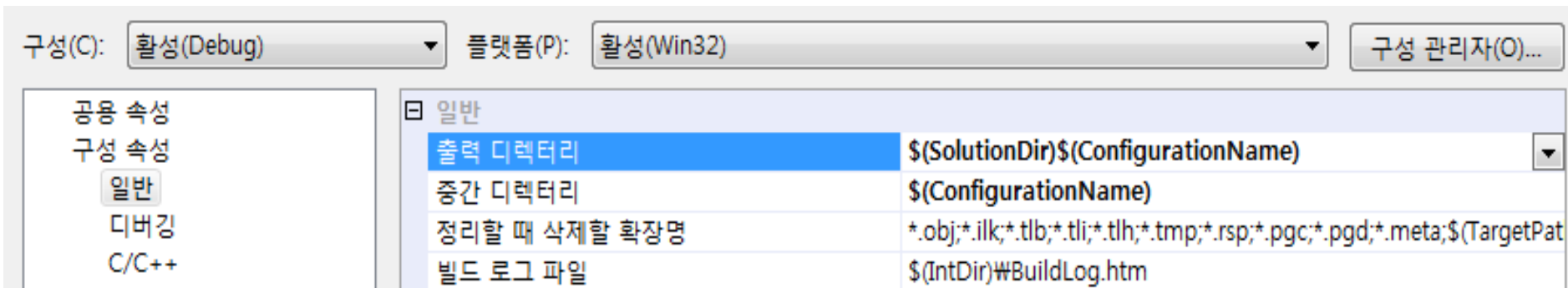


The screenshot shows the 'Properties' window in Visual Studio. The 'General' tab is selected, and the 'Output Directory' property is highlighted with a red circle. The value is set to `..\Bin$(ConfigurationName)`.

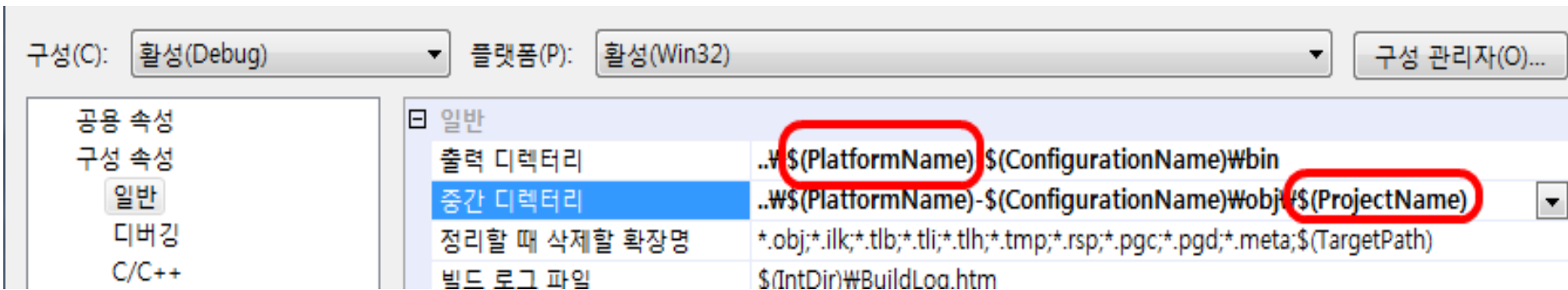
Property Name	Value
출력 디렉터리	..\Bin\$(ConfigurationName)
중간 디렉터리	..\Obj\$(ConfigurationName)
정리할 때 삭제할 확장명	*.obj;*.ilk;*.tlb;*.tli;*.tlh;*.tmp;*.rsp;*.pgc;*.pgd;*.meta;\$(TargetPath)
빌드 로그 파일	\$(IntDir)\BuildLog.htm
상속된 프로젝트 속성 시트	\$(VCInstallDir)VCProjectDefaults\UpgradeFromVC71.vsprops
관리되는 증분 빌드 사용	예

출력 디렉터리는 이렇게

.vcproj 의 출력/중간 디렉터리를 바꾼다.



Better Way



출력 디렉터리를 정할 때

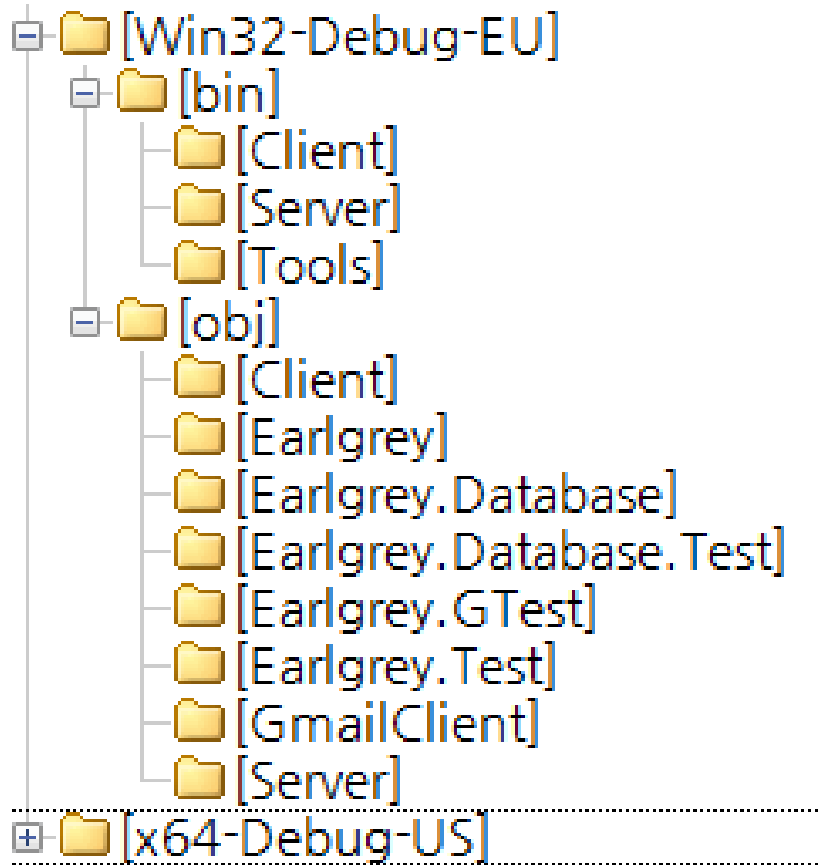
원칙은 이렇다.

출력 파일은 클라이언트/서버군 등 집합 단위로

플랫폼, 구성, 지역 코드 등의 값을 이용해 겹치지 않게

.Obj 파일은 프로젝트 별로

출력 디렉터리의 예



또 다른 흔한 실수

빌드 이벤트

아래 빌드 이벤트는 **실패**로 끝나야 한다.

```
명령줄
ECHO "이 명령어는 반드시 성공"
DIR
ECHO "이 명령어는 반드시 실패"
XCOPY /Y "존재하지 않는 파일.bat" "C:\W"
ECHO "이 명령어는 반드시 성공"
DIR
```

그러나

...

성공한다.



빌드 이벤트는 .bat 일 뿐

빌드 이벤트는 이렇게 해석된다

```
@echo off
ECHO "이 명령어는 반드시 성공"
DIR
ECHO "이 명령어는 반드시 실패"
XCOPY /Y "존재하지 않는 파일.bat" "C:\\"
ECHO "이 명령어는 반드시 성공"
DIR
```

```
if errorlevel 1 goto VCReportError
goto VCEnd
```

```
:VCReportError
```

```
echo Project : error PRJ0019: 도구에서 오류 코드를
반환했습니다. 위치: "빌드 전 이벤트를 수행하고 있습니다..."
exit 1
```

```
:VCEnd
```

실패했는데, 이 아닌 숫자를 반환했다면?
처음 또는 중간 명령이 실패하면?

배치 스크립트를 추방하자

EXIT 코드 확인하기

원칙: 추적 가능

```
@echo off
ECHO "이 명령어는 반드시 성공"
DIR
SET exitCode=%errorlevel%
IF %exitCode% NEQ 0 EXIT / %exitCode%
```

```
ECHO "이 명령어는 반드시 실패"
XCOPY /Y "존재하지 않는 파일.bat" "C:\\"
SET exitCode=%errorlevel%
IF %exitCode% NEQ 0 EXIT / %exitCode%
```

```
ECHO "이 명령어는 반드시 성공"
DIR
SET exitCode=%errorlevel%
EXIT /B %exitCode%
```



MSBuild를 쓰자

간결한 코드

글자 수는 엇비슷, 가독성은 확실!

```
<Target Name="Build Event">  
  —><Message Text="이 명령어는 반드시 성공" />  
  —><Exec Command="Dir" />  
  
  —><Message Text="이 명령어는 반드시 실패" />  
  —><Exec Command="XCOPY /Y &quot;존재하지 않는  
    파일.bat&quot; &quot;C:\&quot;" />  
  
  —><Message Text="이 명령어는 반드시 성공" />  
  —><Exec Command="Dir" />  
</Target>
```

MSBuild를 쓰자

상세한 통합 로그

화려한 콘솔

콘솔과 파일 로그를 동시에

성능 로그

```
관리자: C:\Windows\System32\cmd.exe
2011-04-28 오후 03:10 2,882 msbuild-localbuild-dev.xml (TaskId:1)
2011-05-15 오후 09:14 2,806 msbuild-localbuild.xml (TaskId:1)
2011-05-16 오전 12:42 637 msbuild-recommend-2.xml (TaskId:1)
2011-05-16 오전 12:30 501 msbuild-recommend.xml (TaskId:1)
2011-01-21 오후 12:16 171 MSBuild_Win32.bat (TaskId:1)
2011-01-21 오후 12:16 171 MSBuild_x64.bat (TaskId:1)
2011-01-21 오후 12:16 335 SetEnvironment_Win32.bat (TaskId:1)
2011-01-21 오후 12:16 335 SetEnvironment_x64.bat (TaskId:1)
9개 파일 7,838 바이트 (TaskId:1)
2개 디렉터리 82,972,114,944 바이트 남음 (TaskId:1)
Done executing task "Exec". (TaskId:1)
Done building target "Step1" in project "msbuild-recommend-2.xml". (TargetId:0)
Target "Step2: (TargetId:1)" in file "f:\Downloads\WAAAW새 폴더\msbuild-recommend-2.xml" from project "f:\Downloads\WAAAW새 폴더\msbuild-recommend-2.xml":
Task "Message" (TaskId:2)
이 명령어는 반드시 실패 (TaskId:2)
Done executing task "Message". (TaskId:2)
Task "Exec" (TaskId:3)
명령: (TaskId:3)
XCOPY /Y "존재하지 않는 파일.bat" "C:\W" (TaskId:3)
0개 파일이 복사되었습니다. (TaskId:3)
파일을 찾을 수 없습니다-존재하지 않는 파일.bat (TaskId:3)
f:\Downloads\WAAAW새 폴더\msbuild-recommend-2.xml(11,3): error MSB3073: "XCOPY /Y
"
존재하지 않는 파일.bat" "C:\W"" 명령이 종료되었습니다<코드: 4>.
Done executing task "Exec" -- FAILED. (TaskId:3)
Done building target "Step2" in project "msbuild-recommend-2.xml" -- FAILED.: (
TargetId:1)
Done Building Project "f:\Downloads\WAAAW새 폴더\msbuild-recommend-2.xml" (Build ta
rget<s>) -- FAILED.

Project Performance Summary:
206 ms f:\Downloads\WAAAW새 폴더\msbuild-recommend-2.xml 1 calls
206 ms Build 1 calls

Target Performance Summary:
80 ms Step1 1 calls
95 ms Step2 1 calls

Task Performance Summary:
5 ms Message 2 calls
159 ms Exec 2 calls

Build FAILED.

"f:\Downloads\WAAAW새 폴더\msbuild-recommend-2.xml" (Build target) (1) ->
(Step2 target) ->
f:\Downloads\WAAAW새 폴더\msbuild-recommend-2.xml(11,3): error MSB3073: "XCOPY
/Y
"존재하지 않는 파일.bat" "C:\W"" 명령이 종료되었습니다<코드: 4>.

0 Warning(s)
1 Error(s)

Time Elapsed 00:00:00.20
f:\Downloads\WAAAW새 폴더>
```

MSBuild를 쓰자

확장 기능 개발의 용이함

[MSBuild Community Tasks](#)

[MSBuild EarlGrey Tasks](#)

이를테면

FTP/SFTP, 서브버전, 이메일, 압축, RoboCopy
파일시스템, 공유 폴더, SQL Server, 레지스트리

님아, EXIT_FAILURE 좀

종료 코드가 없다면 실패해도 그런 줄 모른다

```
FILE * fp = fopen(szFilename, "wt");
if(fp)
{
    fprintf(fp, "[CRC]\n");

    for(int i = 0; i < eFile_Max; i++)
        fprintf(fp, "%s = %u\n", g_tHashFiles[i].m_szName, g_tHashFiles[i].m_ulHash);

    fclose(fp);
    printf("%s created.", szFilename);
}
```

오류 메시지만 출력하고
종료 코드는 반환 안 한다.

Better Way



```
FILE * fp = fopen(szFilename, "wt");
if(fp == NULL)
{
    printf("Error: failed to create file %s");
    return 103;
}

fprintf(fp, "[CRC]\n");

for(int i = 0; i < eFile_Max; i++)
    fprintf(fp, "%s = %u\n", g_tHashFiles[i].m_szName, g_tHashFiles[i].m_ulHash);

fclose(fp);

printf("%s created.", szFilename);
```

```
<!--
void _tmain(int argc, _TCHAR* argv[])
{
    std::wcerr << L"오류 발생";
    return;
}
-->
<Target Name="NoExitCode">
    <Message Text="실패 상황인데도 성공 처리된다" />
    <Exec Command="ExitCodeTest-NoExitCode.exe" />
</Target>

<!--
int _tmain(int argc, _TCHAR* argv[])
{
    std::wcerr << L"오류 발생";
    return EXIT_FAILURE;
}
-->
<Target Name="NoExitCode">
    <Message Text="0 이 아닌 값은 오류 처리된다" />
    <Exec Command="ExitCodeTest-ExitFailure.exe" />
</Target>
```

리소스 빌드

수 기가짜리를 어느 세월에 빌드하나?



바뀐 리소스만 빌드하자

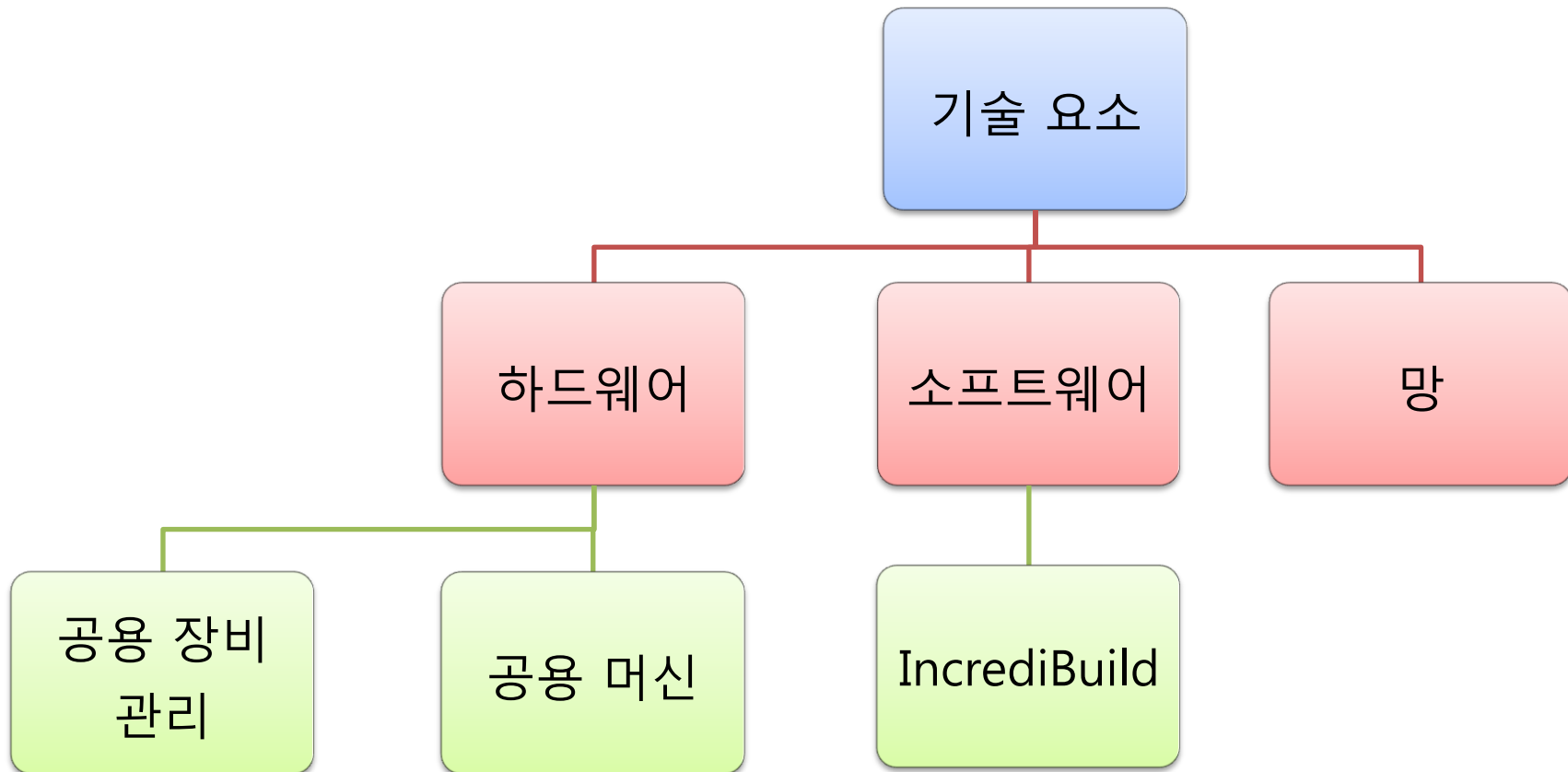
물론 원한다면 전체 다 빌드한다

```
<Target Name="GetUpdatedResourcesList">
  <SvnDiff
    Old="$ (TagOld-Path-Resources) /Game@$ (TagOld-Revision) "
    New="$ (TagNew-Path-Resources) /Game@$ (TagNew-Revision) "
    OldIsBasePath="false"
  >
  <Output TaskParameter="FilesAdded" ItemName="ResourcesAdded" />
  <Output TaskParameter="FilesModified" ItemName="ResourcesModified" />
  <Output TaskParameter="FilesDeleted" ItemName="ResourcesDeleted" />
</SvnDiff>
<!-- 중략 ..... -->
</Target>
```

배포 자동화

하기 전에 인프라 정비부터

개발보다 오래 걸릴 때가 많다



주요 도구

RoboCopy

파일 복사

WinSCP

SFTP

White

UI 자동화

MkLink

심볼릭 링크

배포 환경을 분석해보니

세 가지 배포 방식이 있더라

개발자 빌드

팀 내부 테스트 빌드

출시 빌드

내부 테스트용 배포

공유 폴더에 복사하기

```
<Target Name="CopyClientToRemoteServer">  
  → <BetterRoboCopy  
    → SourceFolder="$(FinalReleaseDir)"  
    → DestinationFolder="\\$(RemoteDir)\Release_$(ShortLocalCode)"  
    → SourceFiles="*"  
    → AllSubdirectories="true"  
    → Options="/R:3"  
    → ExcludeFolders=".svn;_svn"  
    → />  
</Target>
```



배포는 내게 맡겨라

RoboCopy, Why Not XCopy?

병렬 처리

/MT[:n]

네트워크 환경을 고려한 부가 기능

/R:n

다양한 기능

출시 배포

운영 팀에겐 가장 중요한 부분

하지만 그만큼 골 때리는 단계



패키징의 어려움

자동화 지원이 부족한 패키징

명령 줄 / 텍스트 구성 파일

오예~

레지스트리 / 환경 변수

좋아!

구성 파일을 미리 만들기

상대 경로 지정 가능

휴~

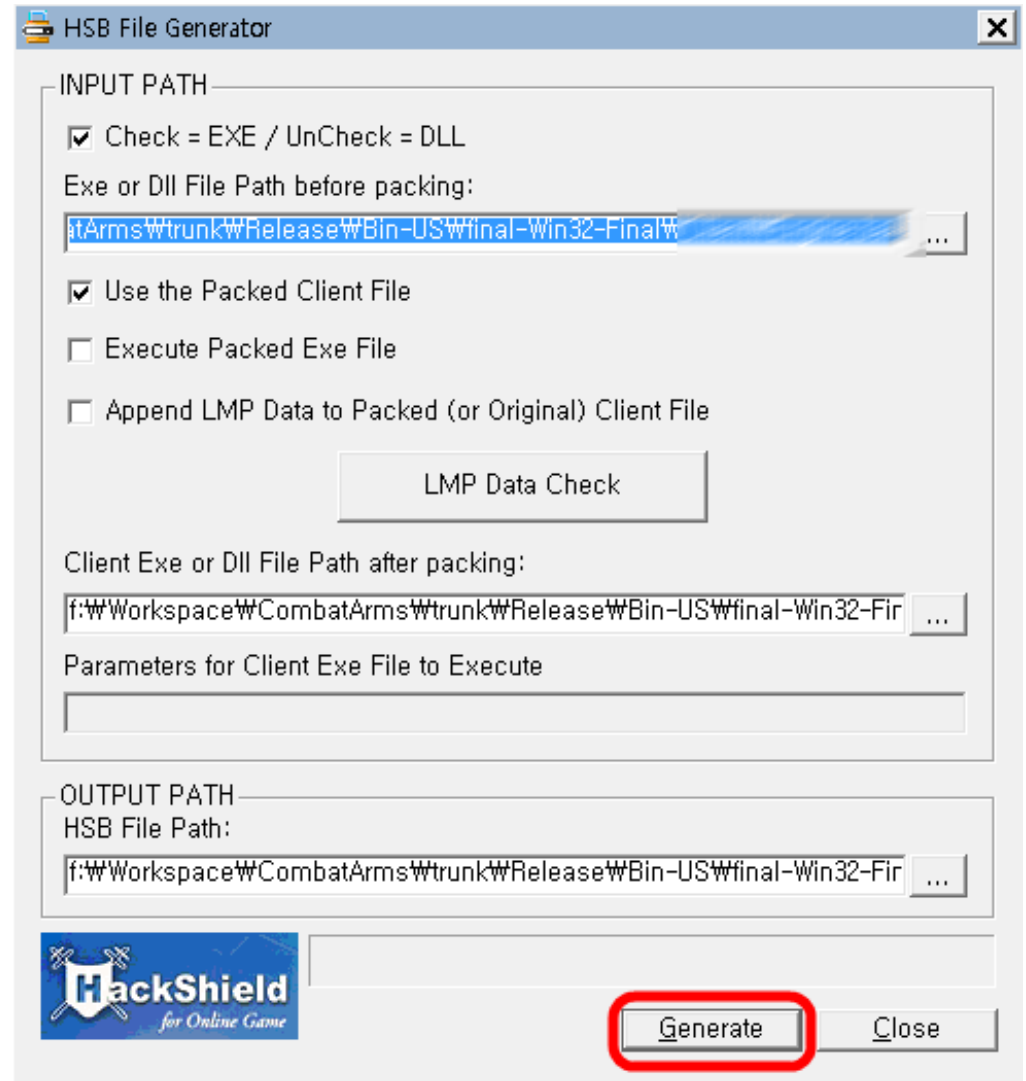
다 안 된다고?

C:\W 는 있겠지



패키징의 어려움

버튼을 눌러다오!



UI 자동화, 별 짓을 다 한다

이럴 때 쓸모 있는 라이브러리 [White](#)

```
using (Window mainWindow = FindWindow("HSB File Generator"))
{
    SearchCriteria fProgressBar = SearchCriteria.ByControlType(ControlType.ProgressBar);
    SearchCriteria fGenerate = SearchCriteria.ByText("Generate");
    SearchCriteria fClose = SearchCriteria.ByText("Close");

    ProgressBar cProgressBar = mainWindow.Get<ProgressBar>(fProgressBar);
    Button cGenerate = mainWindow.Get<Button>(fGenerate);
    Button cClose = mainWindow.Get<Button>(fClose);

    cGenerate.Focus();
    cGenerate.RaiseClickEvent();

    DateTime whenGenerateButtonClicked = DateTime.Now;
    while (true)
    {
        if ((DateTime.Now - whenGenerateButtonClicked) > DefaultExpireDuration)
        {
            Console.Error.WriteLine("오류: HSB File Generator의 Generate 버튼을 눌렀는데도 진행이 전혀 안 됨!");
            return 20;
        }

        if (cProgressBar.Value == cProgressBar.Maximum)
        {
            break;
        }

        Button warningWindowYesButton = mainWindow.Get<Button>(SearchCriteria.ByText("예(Y)"));
        if (warningWindowYesButton != null)
        {
            warningWindowYesButton.Focus();
            warningWindowYesButton.RaiseClickEvent();
        }
    }

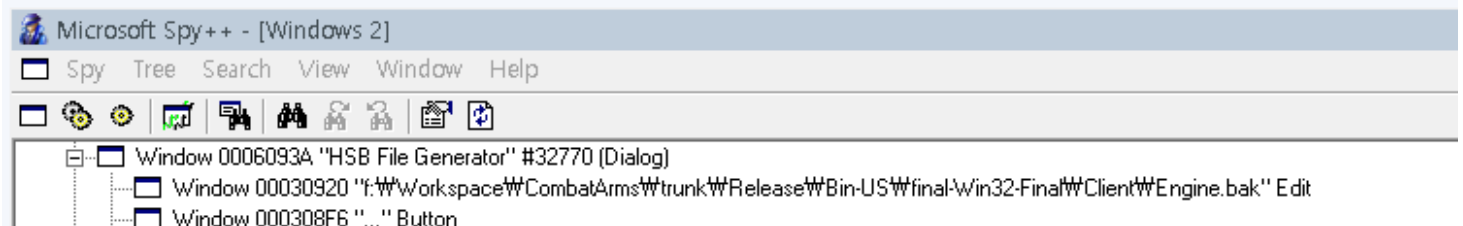
    Button informationWindowOkButton = FindButton(mainWindow, "확인", TimeSpan.FromMinutes(2));
    if (informationWindowOkButton != null)
    {
        informationWindowOkButton.Focus();
        informationWindowOkButton.RaiseClickEvent();
    }

    cClose.RaiseClickEvent();
}
```

UI 자동화, 별 짓을 다 한다

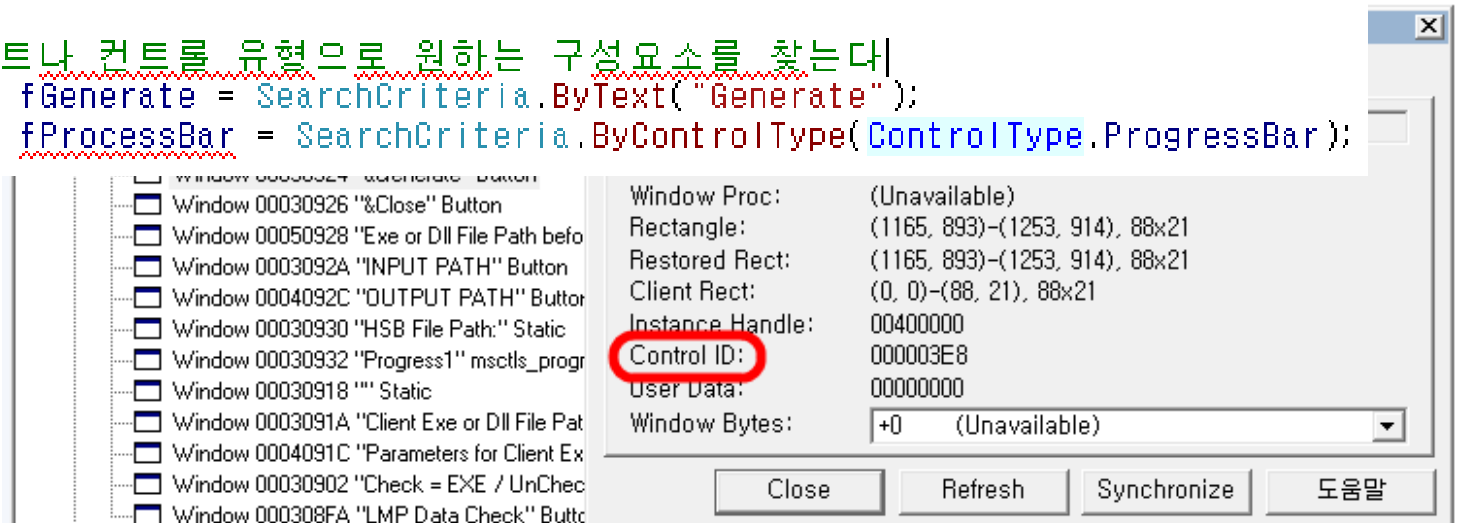
AutomationId에 의존하지 말 것

고유 값이 아니다



```
// Automation Id 는 고유 값이 아니고 바뀌기도 하므로 가급적 쓰지 말자  
// SearchCriteria fGenerate = SearchCriteria.ByAutomationId("1000");
```

```
// 가급적 텍스트나 커트를 유형으로 원하는 구성요소를 찾는다  
SearchCriteria fGenerate = SearchCriteria.ByText("Generate");  
SearchCriteria fProgressBar = SearchCriteria.ByControlType(ControlType.ProgressBar);
```



For Help, press F1

퍼블리셔에게 파일 주기

SFTP 서버에 업로드하기

```
<ItemGroup>
  → <RemoteFolder Include="$(FinalReleaseDir) ">
  →   → <CreateRemoteFolderKeyName>true</CreateRemoteFolderKeyName>
  →   → <RemoteDirKeyName>./$(ShortLocalCode)/$(Version)</RemoteDirKeyName>
  → </RemoteFolder>
</ItemGroup>

<Target Name="CopyClientToRemoteServer">
  → <WinScpUpload
  →   → Files="@ (RemoteFolder) "
  →   → HostName="ftp.publisher.com"
  →   → ProtocolSftp="true"
  →   → UserName="MyId"
  →   → Password="MyPassword"
  →   → HostKey="MyHostKey"
  → />
</Target>
```

지속적인 개선



소회

진짜 장애물은 기술이 아니다

공감대!

조직 차원의 제안

빌드 엔지니어란 직군이 있나요?

THANK YOU!

