

PA Program

: *Property Analysis(Adding) Program*

ISL / 강한솔

Index

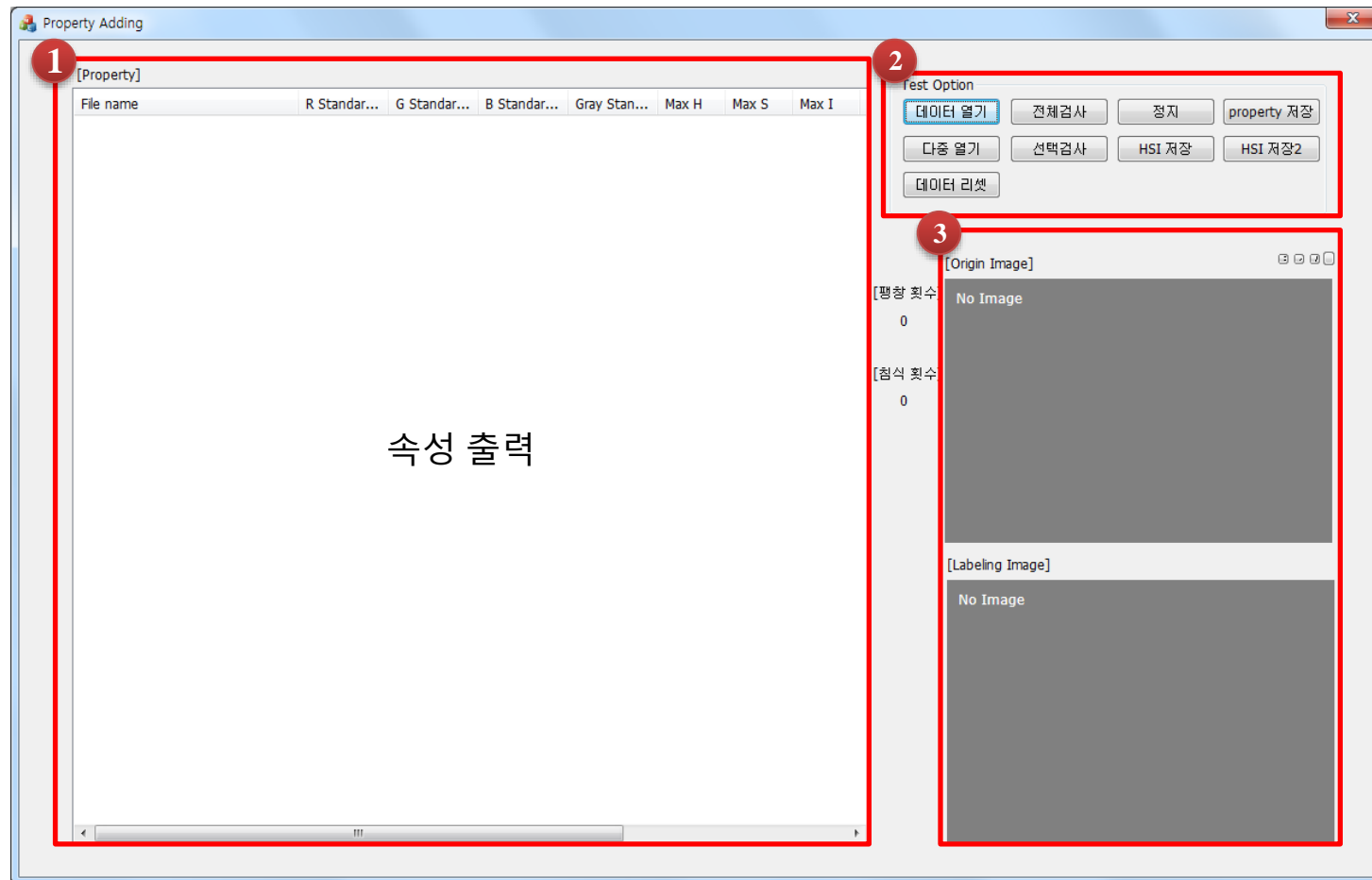
- ✓ Introduction
- ✓ Old PA Program vs. New PA Program
- ✓ Property
- ✓ Function
- ✓ Demo

*Appendix

- ✓ Subversion
- ✓ STL Vector

Introduction

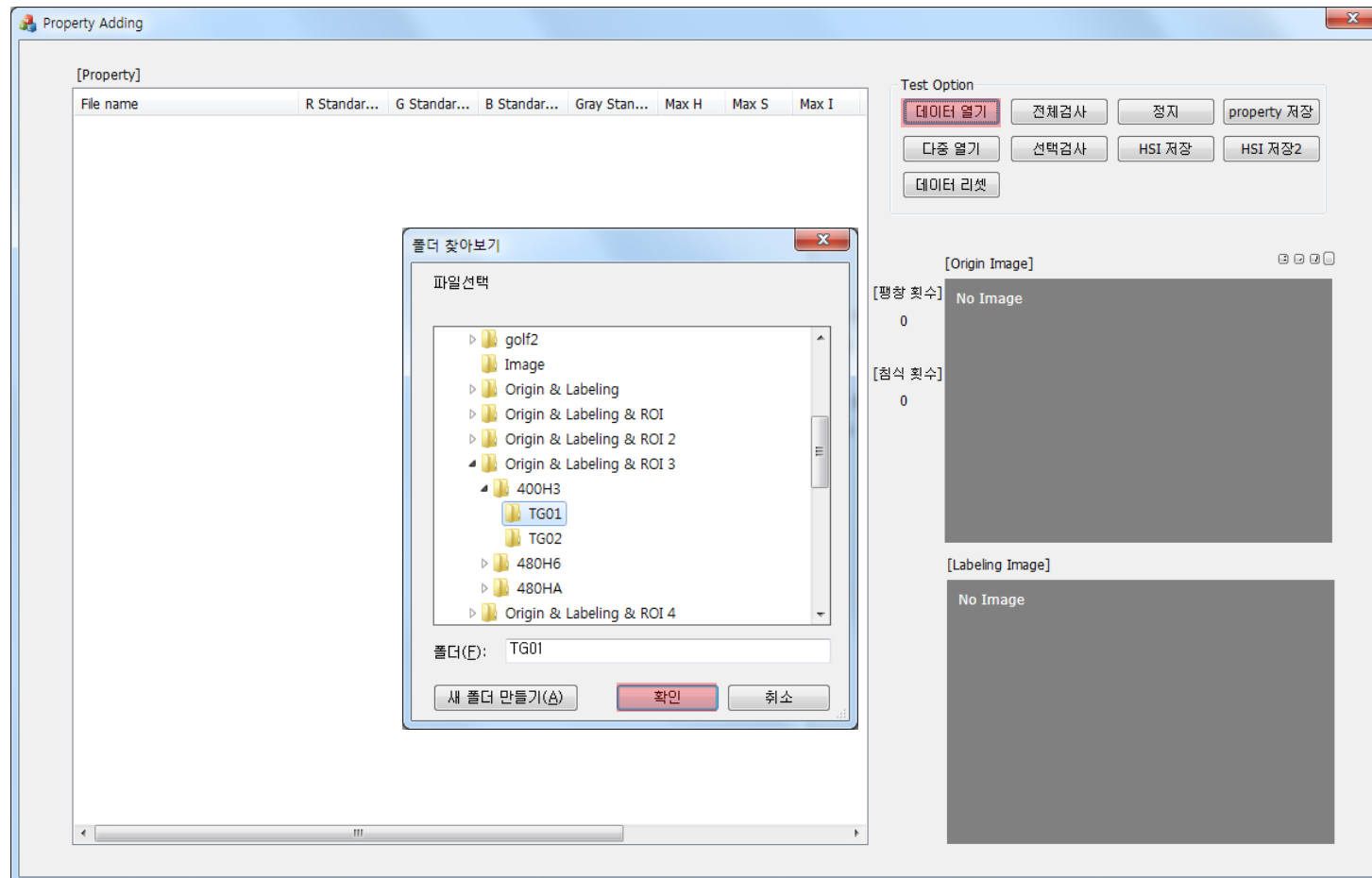
❖ Structure of PA



Introduction

❖ PA usage

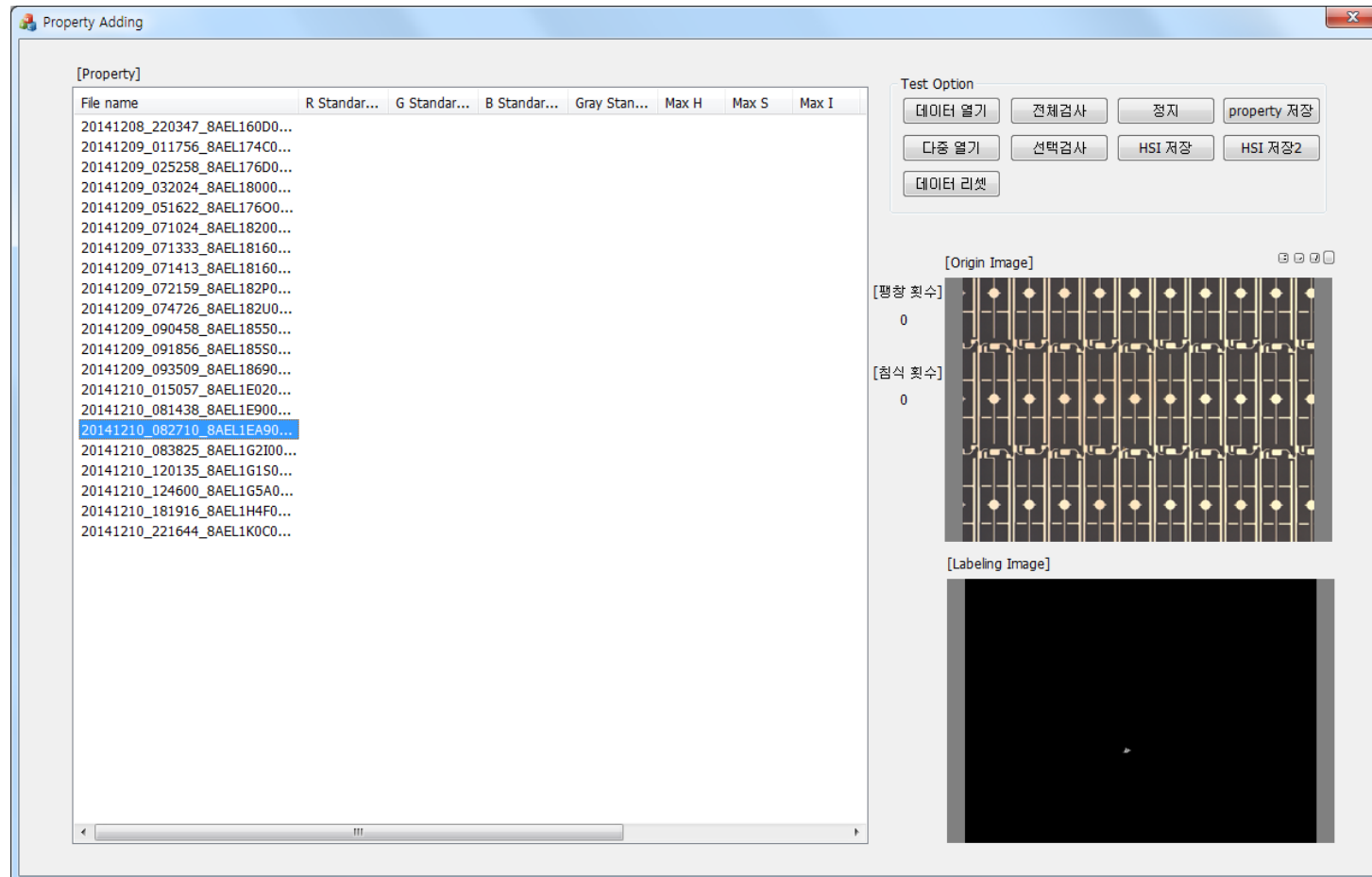
- 데이터 열기



Introduction

❖ PA usage

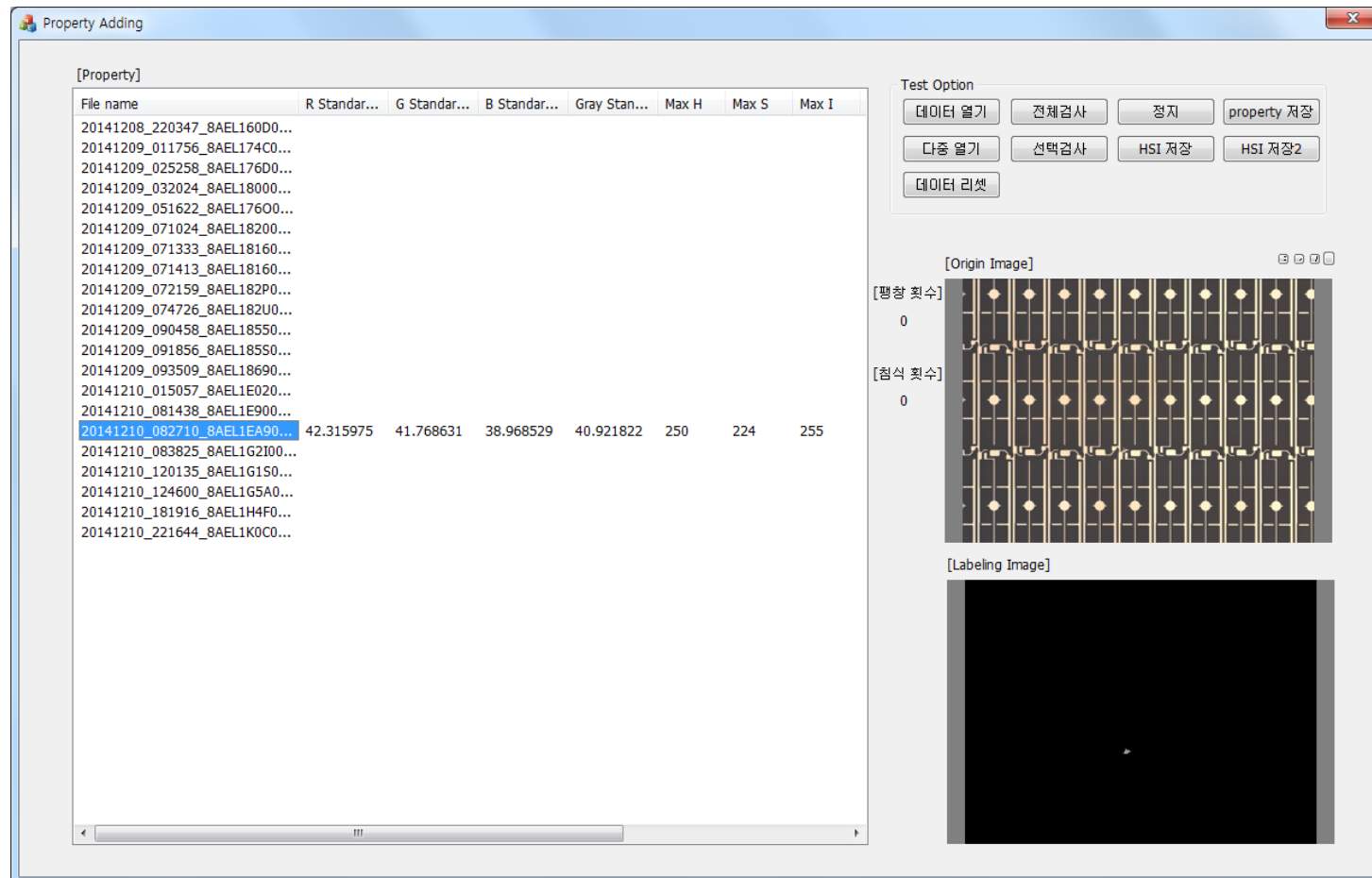
- 데이터 열기



Introduction

❖ PA usage

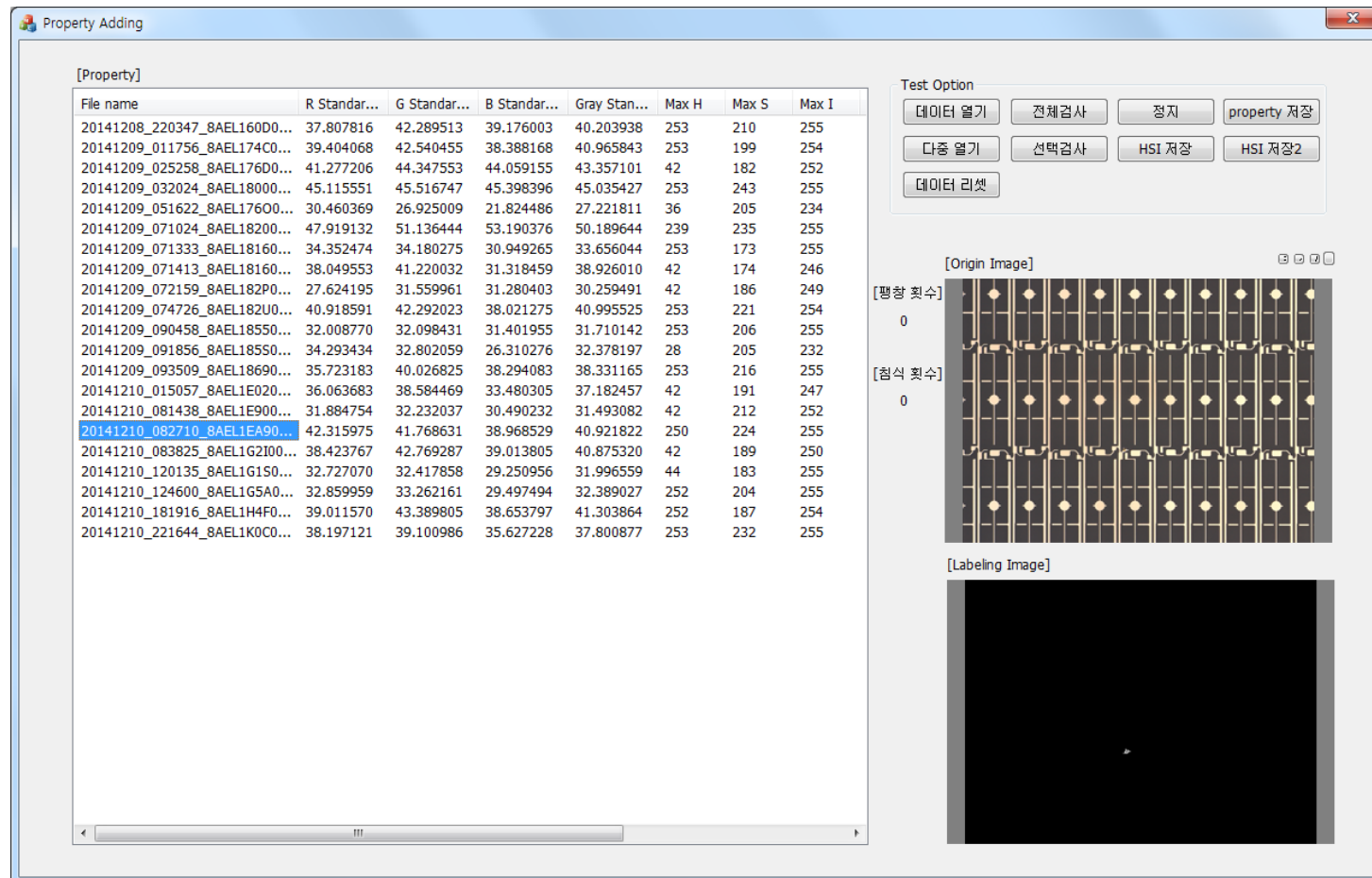
- 선택 검사



Introduction

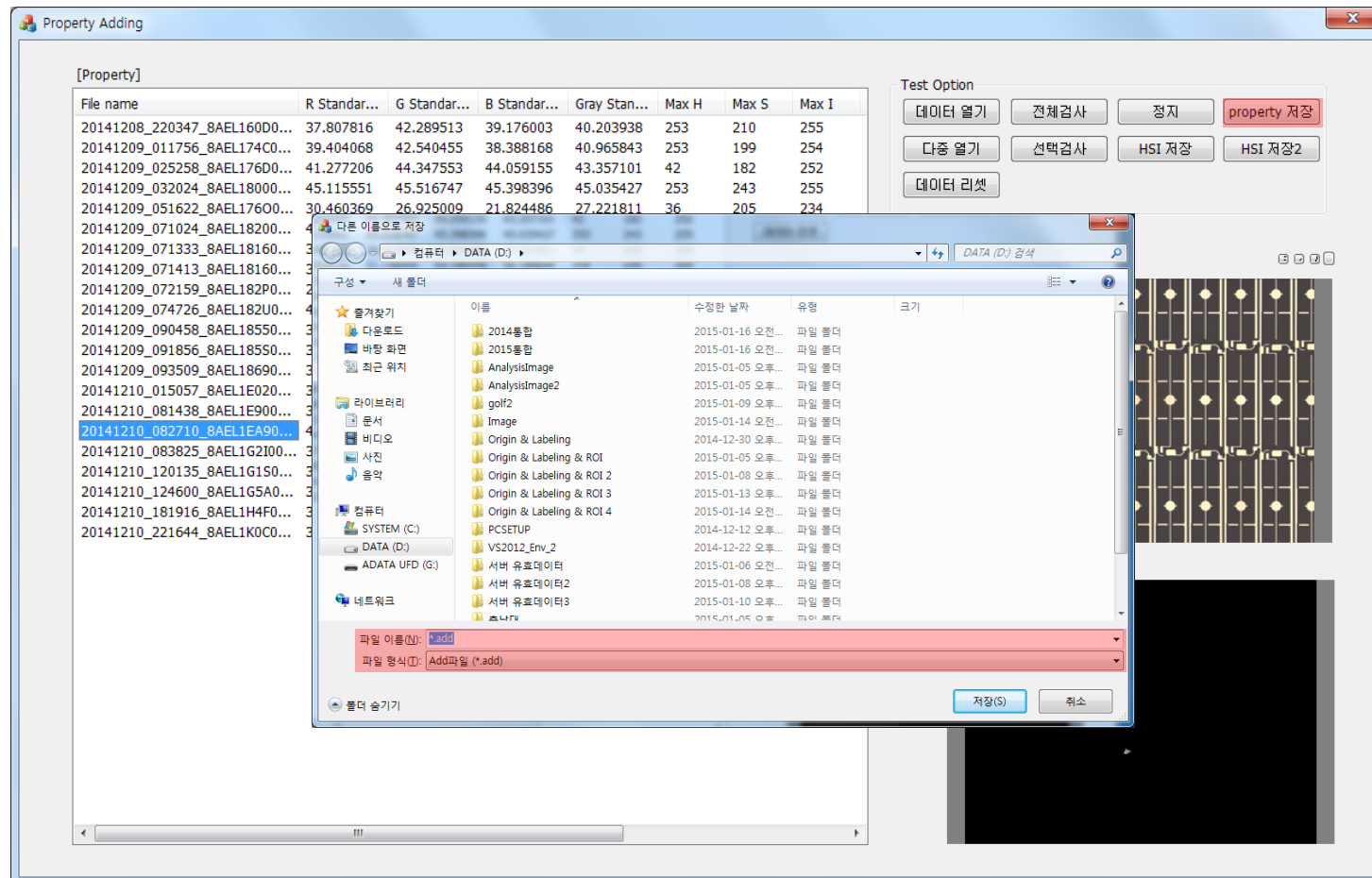
❖ PA usage

- 전체 검사



◆ PA usage

- Property 저장



◆ PA usage

- 
- Test.add



Old PA Program vs. New PA Program

❖ Old PA Program

The screenshot displays the 'Property Adding' window of the Old PA Program. It features a table of RGB Standard Deviation Results and a control panel for test options and image processing.

[RGB Standard Deviation Results]

File name	R Standard Deviation	G Standard Deviation	B Standard Deviation
20141208_123250_8AEL1400014_5_10x...	29.273886	28.644464	22.844805
20141208_220347_8AEL160D005_1_10x...	37.807816	42.289513	39.176003
20141209_011756_8AEL174C006_1_10x...	39.404068	42.540455	38.388168
20141209_025258_8AEL176D005_1_10x...	41.277206	44.347553	44.059155
20141209_032024_8AEL1800005_1_10x...	45.115551	45.516747	45.398396
20141209_051622_8AEL1760007_1_10x...	35.609718	37.775326	34.391857
20141209_060604_8AEL180Z006_3_10x...	40.108841	43.786068	40.943470
20141209_071024_8AEL1820007_1_10x...	47.919132	51.136444	53.190376
20141209_071333_8AEL1816005_2_10x...	34.352474	34.180275	30.949265
20141209_071413_8AEL1816006_3_10x...	38.049553	41.220032	31.318459
20141209_072159_8AEL182P006_3_10x...	27.624195	31.559961	31.280403
20141209_074726_8AEL182U006_3_10x...	40.918591	42.292023	38.021275
20141209_090458_8AEL1855007_1_10x...	32.008770	32.098431	31.401955
20141209_091856_8AEL185S008_1_10x...	34.293434	32.802059	26.310276
20141209_093509_8AEL1869007_2_10x...	35.723183	40.026825	38.294083
20141210_015057_8AEL1E02008_1_10x...	36.063683	38.584469	33.480305
20141210_081438_8AEL1E90006_2_10x...	31.884754	32.232037	30.490232
20141210_082710_8AEL1EA9014_3_10x...	42.315975	41.768631	38.968529
20141210_083825_8AEL1G2I007_1_10x...	38.423767	42.769287	39.013805
20141210_120135_8AEL1G1S006_2_10x...	32.727070	32.417858	29.250956
20141210_124600_8AEL1G5A013_4_10x...	32.859959	33.262161	29.497494
20141210_181916_8AEL1H4F008_1_10x...	39.011570	43.389805	38.653797
20141210_221644_8AEL1K0C013_4_10x...	38.197121	39.100986	35.627228
20141210_231523_8AEL1K10013_1_10x...	36.998146	34.143562	32.090603

Test Option

데이터 열기 전체검사 선택검사 데이터 저장

데이터 리셋

Morphological Operation **ROI**

팽창 침식 모폴로지 리셋 ROI 체크

[Origin Image and Morphology Result]

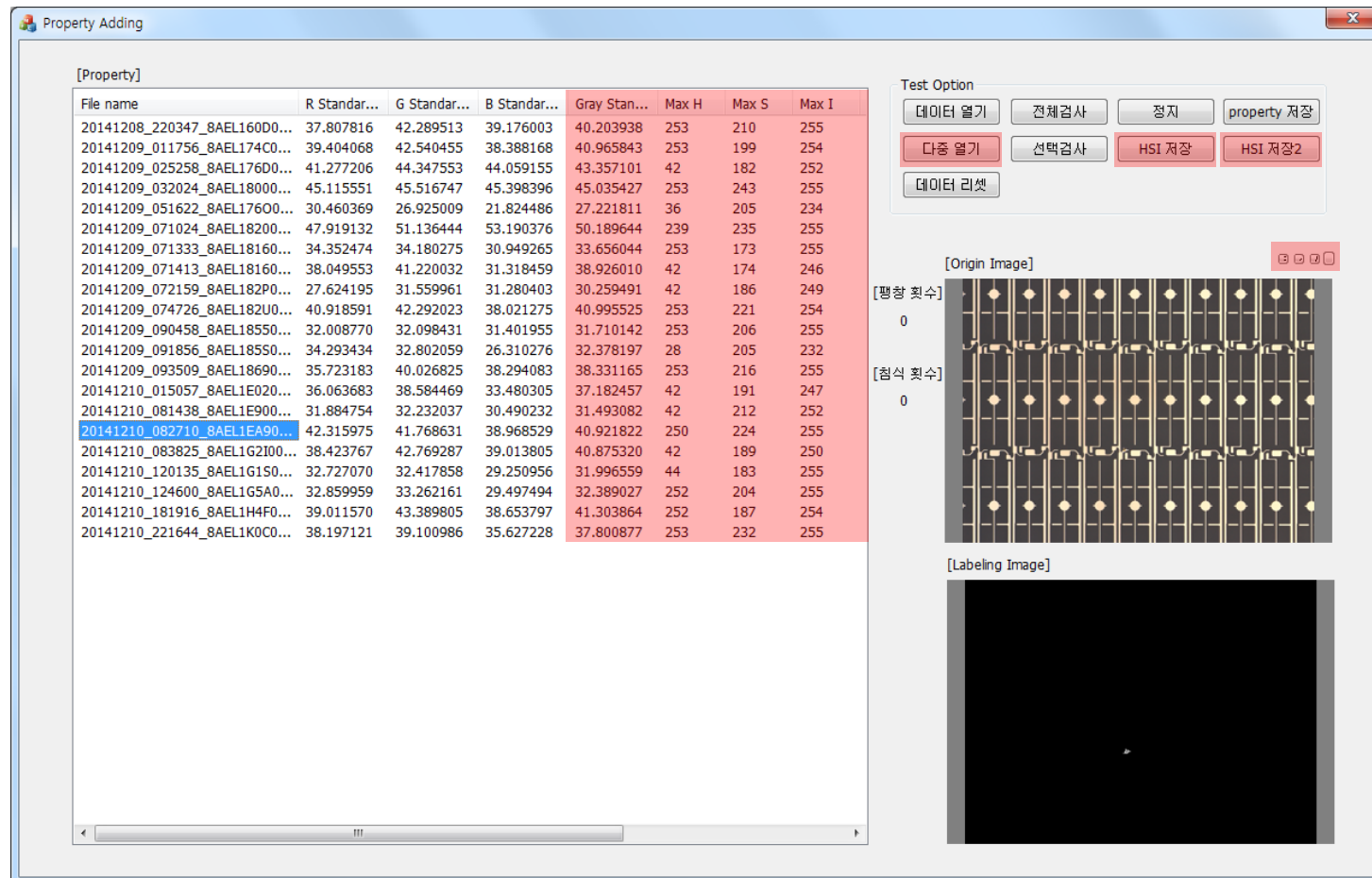
[팽창 횟수] 0

[침식 횟수] 0

[Labeling Image]

Old PA Program vs. New PA Program

❖ New PA Program

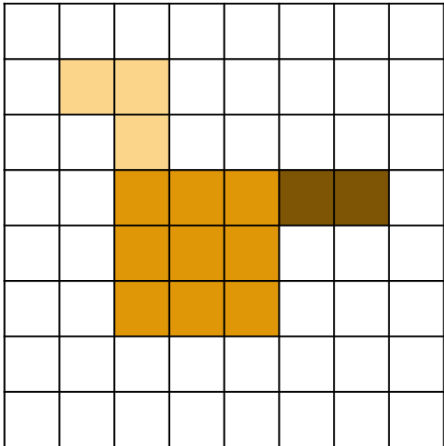



Property


❖ RGB, Gray Standard Deviation


: 각 영역의 평균(R평균, G 평균, B 평균, Gray 평균)보다 높은 값은 무시하고, 평균보다 낮은 값들에 대한 표준편차.

ex) R Standard Deviation



 : R(223), G(151), B(7)

 : R(126), G(85), B(4)

 : R(251), G(213), B(137)

$$Avg(R) = \frac{\sum R}{N} = \frac{(251*3 + 223*9 + 126*2)}{14} = 215.1429$$

$$Std(R) = \sqrt{\frac{\sum (Avg(R) - R)^2}{N}} \text{ (if } Avg(R) > R) = \sqrt{\frac{2 * (215.1429 - 126)^2}{14}}$$

Property

❖ H, S Number

: Hue와 Saturation 히스토그램에서 freq.가 0개인 개수

ex) S number

400H3(TG01) : 20141208_220347_8AEL160D005_1_10x_AOI_TPA070_SP_29__CGJA_B_IN.png

...	193	194	195	196	197	198	199	200	201	202	203	204	205	206	207	208	209	210	211	212	213	214	215	216	217	218	...
...	0	3	1	3	1	3	6	5	5	4	4	5	6	18	26	21	26	3	0	0	0	0	0	0	0	0	...

❖ H, S, I min & max

: 블롭 영역에서 Hue, Saturation, Intensity가 가장 큰 값.

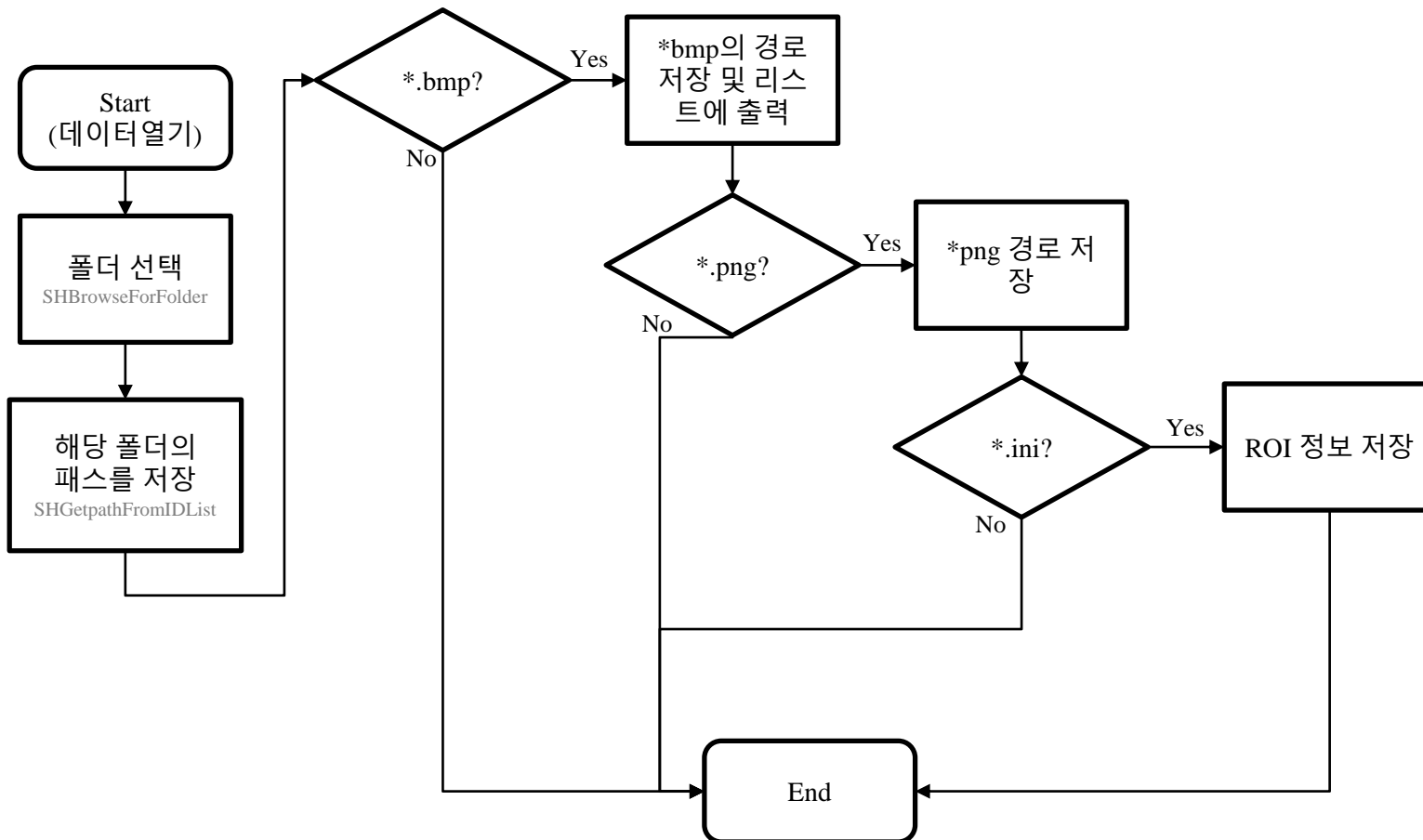
ex) S max

400H3(TG01) : 20141208_220347_8AEL160D005_1_10x_AOI_TPA070_SP_29__CGJA_B_IN.png

...	200	201	202	203	204	205	206	207	208	209	210	211	212	213	214	215	216	217	218	219	220	221	222	223	...	255
...	5	5	4	4	5	6	18	26	21	26	3	0	0	0	0	0	0	0	0	0	0	0	0	0	...	0

Function

❖ 데이터 열기



Function

❖ OnBnClickedButtonOpen()

```
void CImageOpenDlg::OnBnClickedButtonOpen()  
{  
    // TODO: 여기에 컨트롤 알림 처리기 코드를 추가합니다.  
  
    // CFolderPickerDialog dlg("경로", 0, NULL, 0);  
  
    TCHAR szBuffer[MAX_PATH];  
    ZeroMemory(szBuffer, MAX_PATH);  
  
    BROWSEINFO br;  
    ZeroMemory(&br, sizeof(BROWSEINFO));  
  
    br.hwndOwner = m_hWnd;  
    br.lpszTitle = _T("파일선택");  
    br.ulFlags = BIF_NEWDIALOGSTYLE | BIF_EDITBOX | BIF_RETURNONLYFSDIRS;  
    LPITEMIDLIST pitemIdList = SHBrowseForFolder(&br);  
    if(SHGetPathFromIDList(pitemIdList, szBuffer))  
    {  
        /*::SHGetPathFromIDList(pitemIdList, szBuffer);*/  
  
        CFileFind finder;  
        CString fileName;  
        CString path;  
  
        path=szBuffer;
```

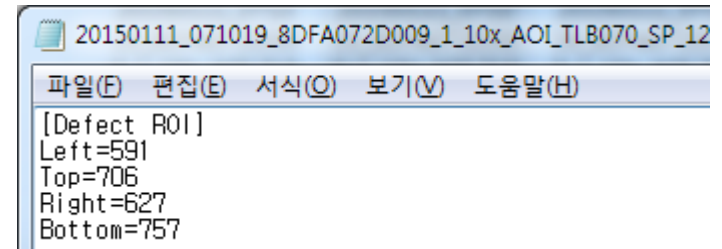
```
////////////////////////////////////////png 파일 path 저장  
        BOOL bWorkOrigin = finder.FindFile(path+".*.*.png");  
        if(bWorkOrigin==0)  
        {  
            AfxMessageBox(_T(".png파일이 존재하지 않습니다."));  
            return;  
        }  
        while(bWorkOrigin)  
        {  
            bWorkOrigin=finder.FindNextFileW();  
            if(finder.IsArchived())  
            {  
                m_ctrlList.InsertItem(m_nOriginNumber,finder.GetFileName());  
                m_vecstrImagePath.push_back(finder.GetFilePath());  
                m_nOriginNumber++;  
            }  
        }  
    }
```

Function

❖ OnBnClickedButtonOpen()

```
////////////////////////////////////ini(R0I) 파일 path 저장
BOOL bWorkRoi = finder.FindFile(path+".***.ini");
if(bWorkRoi==0)
{
    AfxMessageBox(_T("ini파일이 존재하지 않습니다."));
    return;
}

while(bWorkRoi)
{
    bWorkRoi=finder.FindNextFileW();
    if(finder.IsArchived())
    {
        CString strPath = finder.GetFilePath();
        CRect rtRoi;
        rtRoi.left =GetPrivateProfileInt(_T("Defect R0I"),_T("Left") ,0,strPath);
        rtRoi.top =GetPrivateProfileInt(_T("Defect R0I"),_T("Top") ,0,strPath);
        rtRoi.right =GetPrivateProfileInt(_T("Defect R0I"),_T("Right ") ,0,strPath);
        rtRoi.bottom=GetPrivateProfileInt(_T("Defect R0I"),_T("Bottom"),0,strPath);
        m_vecrtRoi.push_back(rtRoi);
    }
}
```



GetPrivateProfileInt(LPCTSTR lpAppName, LPCTSTR lpKeyName, INT nDefault, LPCTSTR lpFileName)

해당 섹션의 이름

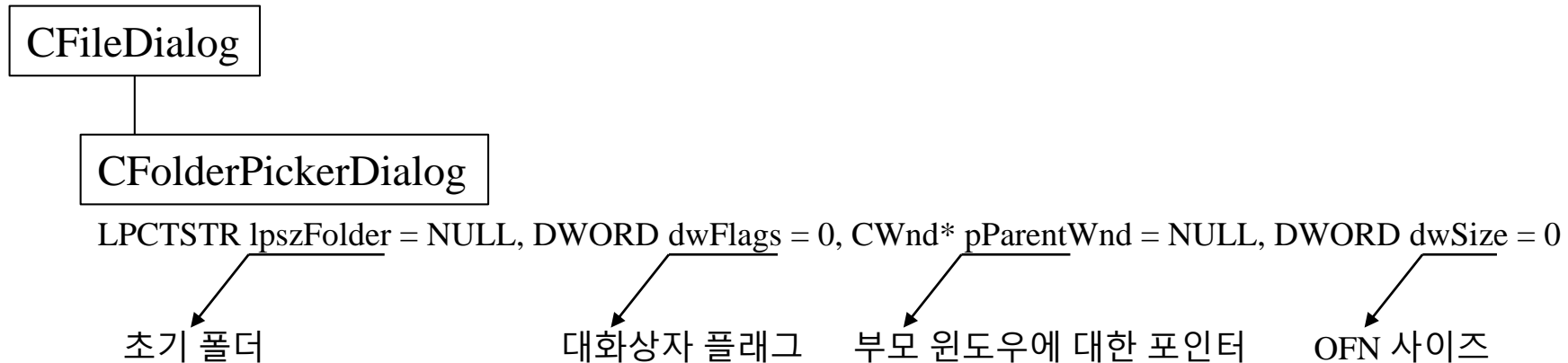
해당 키 이름

키 이름을 찾지 못했을 때 읽어오는 값

ini 파일경로

Function

❖ 다중 열기



[dwFlag]

- OFN_EXPLORER 윈도우 탐색기 스타일로 출력
- OFN_ALLOWMULTISELECT 파일을 한번에 여러개 선택 가능
- OFN_CREATEPROMPT 존재하지 않는 파일명을 입력했을 경우 새로 생성하겠냐는 메시지 박스 출력
- OFN_FILEMUSTEXIST 존재하지 않는 파일명을 입력할 수 없도록 함
- OFN_HIDEREADONLY 읽기 전용 파일은 출력하지 않음
- OFN_LONGNAMES 긴 파일 이름 포맷 지원
- OFN_OVERWRITEPROMPT 존재하는 파일명을 입력했을 경우 덮어쓰겠냐는 메시지 박스 출력
- OFN_PATHMUSTEXIST 이미 존재하는 디렉터리명만을 입력

Function

❖ OnBnClickedButtonMulti()

```
void CImageOpenDlg::OnBnClickedButtonMulti()
{
    // TODO: 여기에 컨트롤 알림 처리기 코드를 추가합니다.
    CString Init;
    Init.Format(_T("D:***"));
    CFolderPickerDialog PicDlg(NULL, OFN_ALLOWMULTISELECT);
    CString file;
    PicDlg.GetOFN().lpstrInitialDir = Init.GetBuffer(_MAX_PATH);
    PicDlg.GetOFN().lpstrFile = file.GetBuffer(Multi*(_MAX_PATH+1)+1);
    PicDlg.GetOFN().nMaxFile = (Multi)*(_MAX_PATH+1)+1;
    CFileFind finder;
    CString path;

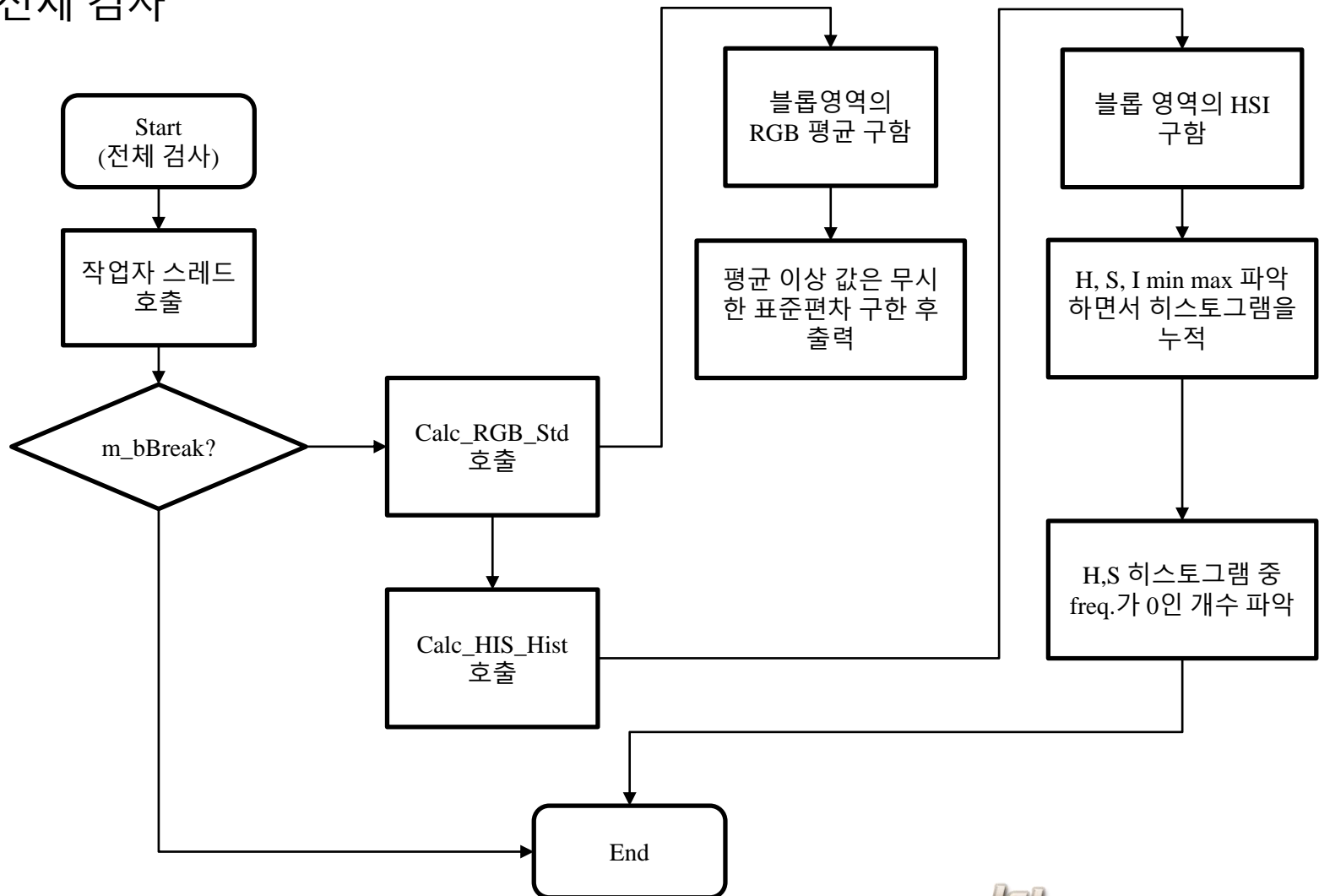
    if(IDOK==PicDlg.DoModal())
    {
        POSITION pos=PicDlg.GetStartPosition();
        while(pos)
        {
            path=PicDlg.GetNextPathName(pos);
        }
    }
}
```

```
////////////////////////////////////png 파일 path 저장.
BOOL bWorkOrigin = finder.FindFile(path+"*.png");
if(bWorkOrigin==0)
{
    AfxMessageBox(_T("*.png파일이 존재하지 않습니다."));
    return;
}
while(bWorkOrigin)
{
    bWorkOrigin=finder.FindNextFileW();
    if(finder.IsArchived())
    {
        m_ctrlList.InsertItem(m_nOriginNumber,finder.GetFileName());
        m_vecstrImagePath.push_back(finder.GetFilePath());
        m_nOriginNumber++;
    }
}
```

```
////////////////////////////////////ini(ROI) 파일 path 저장
BOOL bWorkRoi = finder.FindFile(path+"*.ini");
if(bWorkRoi==0)
{
    AfxMessageBox(_T("ini파일이 존재하지 않습니다."));
    return;
}
while(bWorkRoi)
{
    bWorkRoi=finder.FindNextFileW();
    if(finder.IsArchived())
    {
        CString strPath = finder.GetFilePath();
        CRect rtROI;
        rtROI.left =GetPrivateProfileInt(_T("Defect ROI"),_T("Left") ,0,strPath);
        rtROI.top =GetPrivateProfileInt(_T("Defect ROI"),_T("Top") ,0,strPath);
        rtROI.right =GetPrivateProfileInt(_T("Defect ROI"),_T("Right ") ,0,strPath);
        rtROI.bottom=GetPrivateProfileInt(_T("Defect ROI"),_T("Bottom"),0,strPath);
        m_vecrtROI.push_back(rtROI);
    }
}
```

Function

❖ 전체 검사



Function

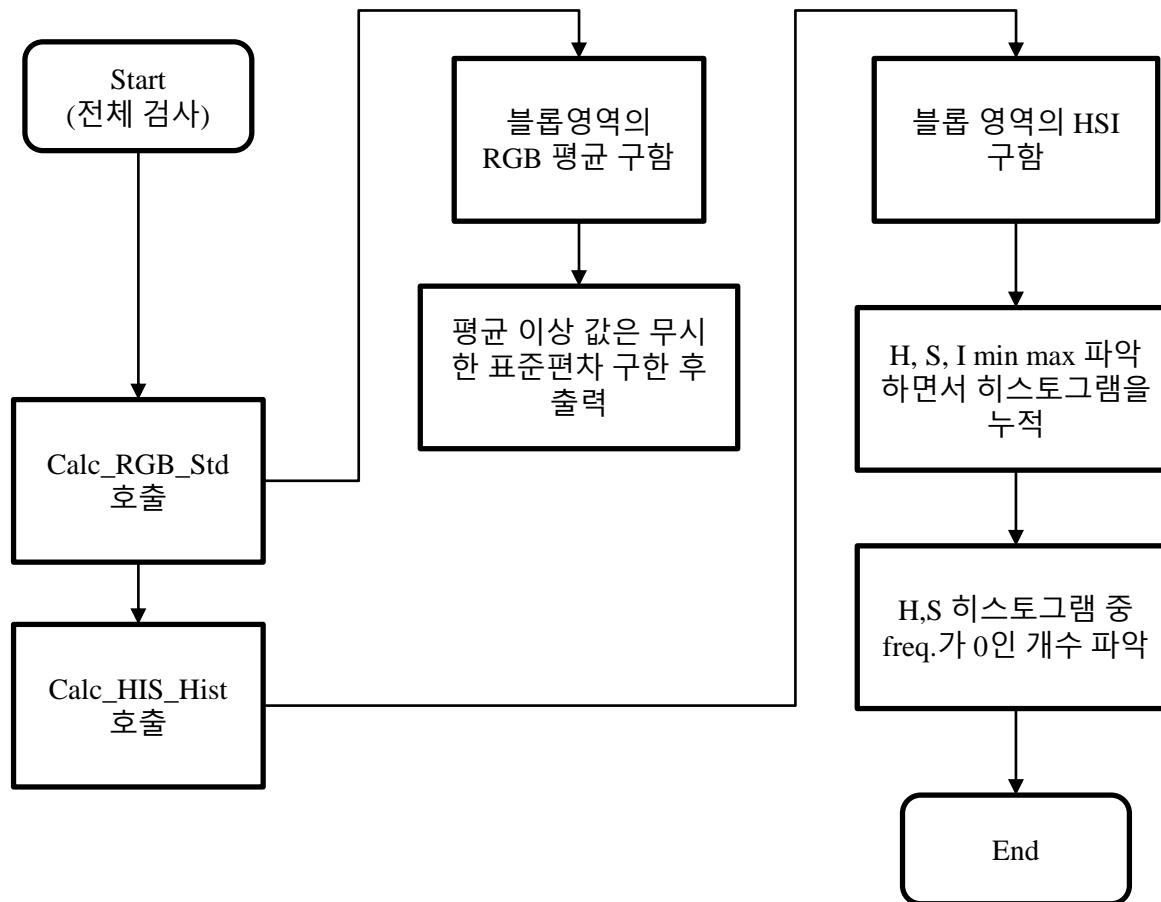
❖ 전체 검사

```
void CImageOpenDlg::OnBnClickedButtonEntire()  
{  
    ////////////////////////////////////////////전체검사/////////////////////////////////////////  
    if(Error());  
    else  
    {  
        m_bBreak=0;  
        //m_vecnHhist.resize(m_nOriginNumber);  
        //m_vecnShist.resize(m_nOriginNumber);  
        CWinThread* pThread = AfxBeginThread(ThreadEntire, this);        //작업자 스레드 호출  
        if(pThread == NULL)  
        {  
            AfxMessageBox(_T("Error : 작업자 스레드 시작을 실패했습니다!!!"));  
        }  
    }  
}
```

```
UINT CImageOpenDlg::ThreadEntire(LPVOID pParam)  
{  
    //전체 검사 스레드  
    CImageOpenDlg *pListDlg = (CImageOpenDlg *)pParam;  
    for(int number=0; number<pListDlg->m_nOriginNumber; number++)  
    {  
        pListDlg->m_pImage->Free();  
        pListDlg->m_pImage->LoadFrom(pListDlg->m_vecstrImagePath[number]);  
  
        pListDlg->m_pImageLabel->Free();  
        pListDlg->m_pImageLabel->LoadFrom(pListDlg->m_vecstrLabelPath[number]);  
  
        pListDlg->Calc_RGB_Std(number, pListDlg->m_pImage, pListDlg->m_pImageLabel, pListDlg->m_vecrtROI[number]);  
        pListDlg->Calc_Gray_Std(number, pListDlg->m_pImage, pListDlg->m_pImageLabel, pListDlg->m_vecrtROI[number]);  
        pListDlg->Calc_HSI_Hist(number, pListDlg->m_pImage, pListDlg->m_pImageLabel, pListDlg->m_vecrtROI[number]);  
  
        //pListDlg->m_pImageView2->SetImage(*pListDlg->m_pImageLabel);        //이미지 출력  
        //pListDlg->m_pImageView->SetImage(*pListDlg->m_pImage);        //이미지 출력  
  
        if(pListDlg->m_bBreak==1) return 0;  
    }  
    return 0;  
}
```

Function

❖ 선택 검사



Function

❖ 선택 검사

```
void CImageOpenDlg::OnBnClickedButtonSelect( )
{
    //////////////////////////////////////////선택검사////////////////////////////////////////
    if(Error( ));
    else
    {
        int nSelectedItem = m_ctrlList.GetNextItem( -1, LVNI_SELECTED );

        if(nSelectedItem>=0)
        {
            Calc_RGB_Std(nSelectedItem,m_pImage, m_pImageLabel,m_vecrtROI[nSelectedItem]);
            Calc_Gray_Std(nSelectedItem,m_pImage, m_pImageLabel,m_vecrtROI[nSelectedItem]);
            Calc_HSI_Hist(nSelectedItem,m_pImage, m_pImageLabel,m_vecrtROI[nSelectedItem]);
        }
        else AfxMessageBox(_T("선택된 파일이 없습니다!!!"));
    }
}
```

Function

❖ Calc_RGB_Std(int number, IV_dataImage_8u_C3 *Image, IV_dataImage_8u_C1 *ImageLabel, CRect ROI)

리스트에서 몇 번째 아이템인지 나타내는 수

Origin(png 이미지)

Labeling(bmp 이미지)

ROI 정보

```
void CImageOpenDlg::Calc_RGB_Std(int number, IV_dataImage_8u_C3 *Image, IV_dataImage_8u_C1 *ImageLabel, CRect ROI)
```

```
{
    float fR=0.f, fG=0.f, fB=0.f;
    float stdR = 0.f, stdG=0.f, stdB=0.f;
    int area=0;
    CString strR, strG, strB;
    long nColorByte = Image->GetBpp()/8;
    long nLabel=255;
    BYTE *PtLabel=ImageLabel->GetMem();

    CSize Origin = Image->GetSize();

    BYTE* pbyColor=Image->GetRawMem();

    long nX=0, nY=0;
    long nOffset1 = 0, nOffset2=0;

    for(ROI.top; nY<ROI.bottom; nY++)
    {
        for(nX=ROI.left; nX<ROI.right; nX++)
        {
            nOffset1 = nY * Origin.cx + nX;
            nOffset2 = (nY * Origin.cx + nX)*nColorByte;
            if(PtLabel[nOffset1] == nLabel)
            {
                fR+=pbyColor[nOffset2+2];
                fG+=pbyColor[nOffset2+1];
                fB+=pbyColor[nOffset2];
                area++;
            }
        }
    }

    fR=fR/area;
    fG=fG/area;
    fB=fB/area;
```

```
    for(nY=ROI.top; nY<ROI.bottom-1; nY++)
    {
        for(nX=ROI.left; nX<ROI.right; nX++)
        {
            nOffset1 = nY * Origin.cx + nX;
            nOffset2 = (nY * Origin.cx + nX)*nColorByte;
            if(PtLabel[nOffset1] == nLabel)
            {
                if(pbyColor[nOffset2+2]<fR) stdR+=pow(pbyColor[nOffset2+2]-fR, 2.f);
                if(pbyColor[nOffset2+1]<fG) stdG+=pow(pbyColor[nOffset2+1]-fG, 2.f);
                if(pbyColor[nOffset2]<fB) stdB+=pow(pbyColor[nOffset2]-fB, 2.f);
            }
        }

        stdR = sqrt(stdR/area);
        stdG = sqrt(stdG/area);
        stdB = sqrt(stdB/area);
        // CString str;
        strR.Format(_T("%f"),stdR);
        strG.Format(_T("%f"),stdG);
        strB.Format(_T("%f"),stdB);

        m_ctrlList.SetItemText(number,1,strR);
        m_ctrlList.SetItemText(number,2,strG);
        m_ctrlList.SetItemText(number,3,strB);
        //m_ctrlList.SetItem(number,1,LVIF_TEXT,strR,0,0,0,0);
        //m_ctrlList.SetItem(number,2,LVIF_TEXT,strG,0,0,0,0);
        //m_ctrlList.SetItem(number,3,LVIF_TEXT,strB,0,0,0,0);
    }
}
```

Function

❖ Calc_HSI_Hist(int number, IV_dataImage_8u_C3 *Image, IV_dataImage_8u_C1 *ImageLabel, CRect ROI)

리스트에서 몇 번째 아이템인지 나타내는 수

Origin(png 이미지)

Labeling(bmp 이미지)

ROI 정보

$$H = \begin{cases} H1 & \text{if } B \leq G \\ 360^\circ - H1 & \text{if } B > G \end{cases}$$

$$H1 = \cos^{-1} \left\{ \frac{0.5[(R - G) + (R - B)]}{\sqrt{(R - G)^2 + (R - B)(G - B)}} \right\}$$

$$S = \frac{\text{Max}(R, G, B) - \text{Min}(R, G, B)}{\text{Max}(R, G, B)}$$

$$I = \frac{\text{Max}(R, G, B)}{255}$$

Function

❖ Calc_HSI_Hist(int number, IV_dataImage_8u_C3 *Image, IV_dataImage_8u_C1 *ImageLabel, CRect ROI)

```
num=((fR-fG)+(fR-fB))/2.f;
den=sqrt((fR-fG)*(fR-fG)+(fR-fB)*(fG-fB));
if(fR==fG && fG==fB)                                     //Gray scale일 때
{
    nH=0;
    nS=0;
}
else
{
    if(fB>fG)
    {
        nH=(360.f-acos(num/den)*180.f/Pi)*(255.f/360.f);    //H구함 0~255 range
    }
    else
    {
        nH=(acos(num/den)+180.f/Pi)*(255.f/360.f);
    }
}
fRgbmin= min(fRgbmin,fR);
fRgbmin= min(fRgbmin,fG);
fRgbmin= min(fRgbmin,fB);

nS=(1.f-(3.f*fRgbmin/(fR+fG+fB)))*255.f;                    //S구함
nI=(fR+fG+fB)/3.f;                                          //I구함

nHmax=max(nHmax,nH);                                       //HSI max min 구함
nSmax=max(nSmax,nS);
nImax=max(nImax,nI);

nHmin=min(nHmin,nH);
nSmin=min(nSmin,nS);
nImin=min(nImin,nI);

histH[nH]++;
histS[nS]++;
histI[nI]++;                                              //히스토그램 저장
```

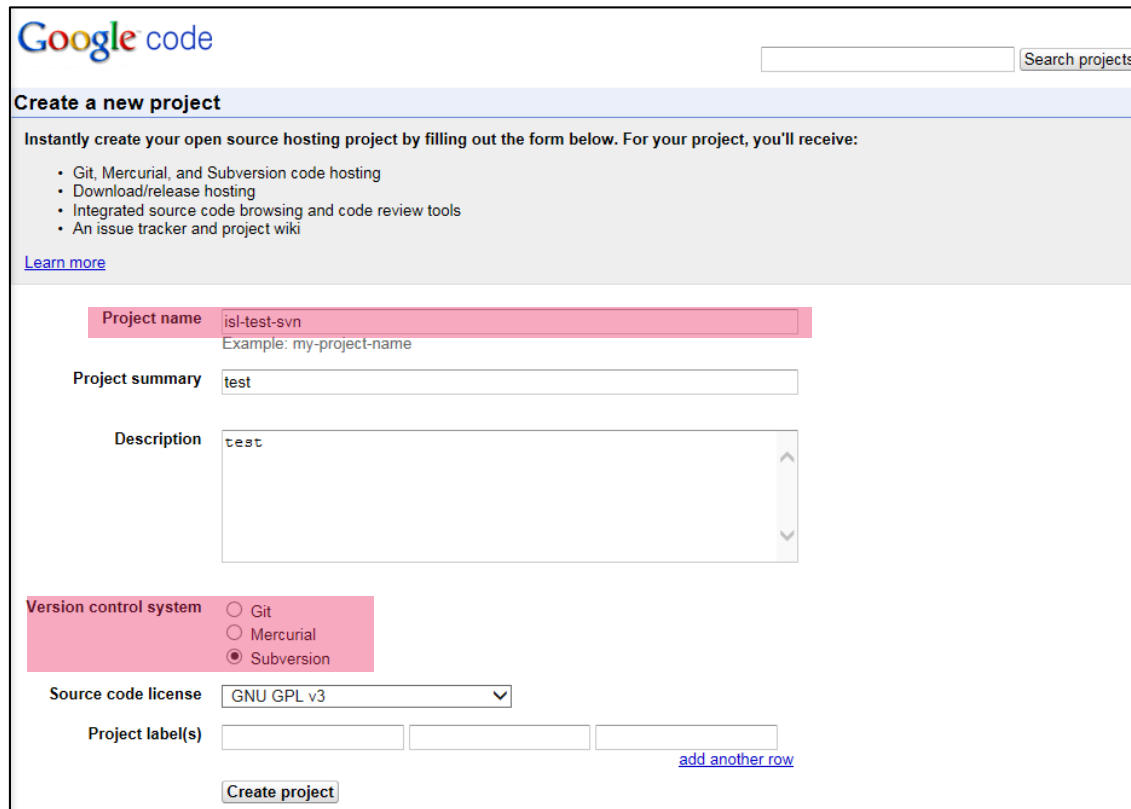
Demo



*Appendix - Subversion

❖ Subversion(SVN)

: 프로그램을 버전 별로 관리하는 시스템.



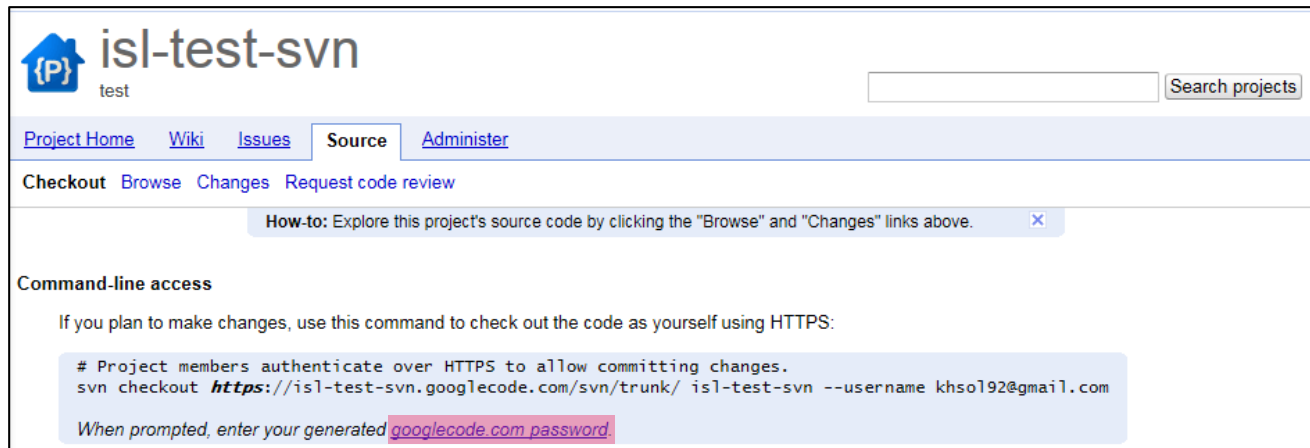
The screenshot shows the 'Create a new project' page on Google Code. The form is filled out with the following information:

- Project name:** isl-test-svn (Example: my-project-name)
- Project summary:** test
- Description:** test
- Version control system:** Subversion (selected with a radio button, alongside Git and Mercurial)
- Source code license:** GNU GPL v3 (selected from a dropdown menu)
- Project label(s):** Three empty text boxes with a link 'add another row' below them.

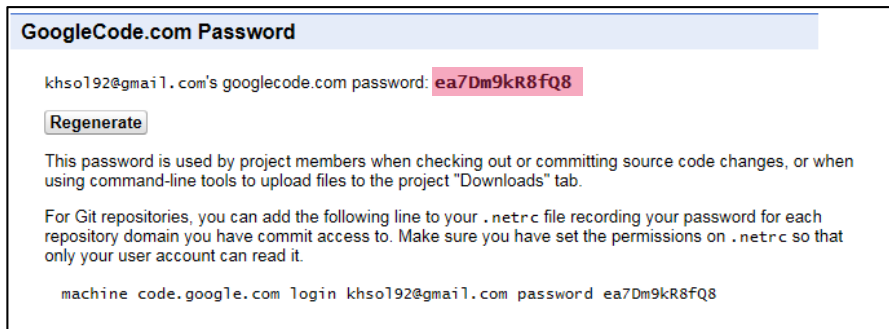
The 'Create project' button is at the bottom of the form.

*Appendix - Subversion

❖ Subversion(SVN)



The screenshot shows the Google Code project page for 'isl-test-svn'. At the top, there's a project icon and name 'isl-test-svn' with a 'test' label. A search bar is on the right. Below the project name are tabs for 'Project Home', 'Wiki', 'Issues', 'Source' (which is selected), and 'Administer'. Under the 'Source' tab, there are links for 'Checkout', 'Browse', 'Changes', and 'Request code review'. A 'How-to' message says: 'Explore this project's source code by clicking the "Browse" and "Changes" links above.' Below this, the 'Command-line access' section provides instructions: 'If you plan to make changes, use this command to check out the code as yourself using HTTPS:'. It then shows a code block: '# Project members authenticate over HTTPS to allow committing changes. svn checkout https://isl-test-svn.googlecode.com/svn/trunk/ isl-test-svn --username khso192@gmail.com'. A note below the code block says: 'When prompted, enter your generated googlecode.com password.'

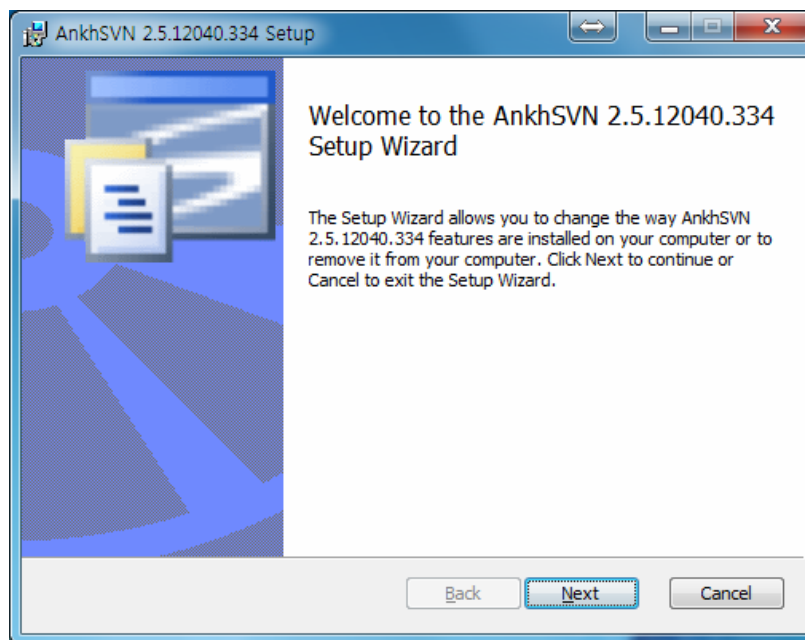
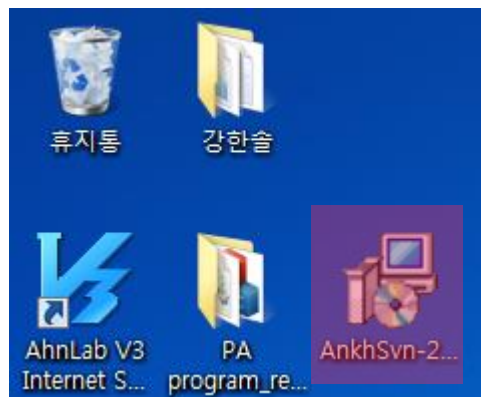


The screenshot shows the 'GoogleCode.com Password' page. It displays the user 'khso192@gmail.com's googlecode.com password: ea7Dm9kR8fQ8'. There is a 'Regenerate' button. Below the password, a message states: 'This password is used by project members when checking out or committing source code changes, or when using command-line tools to upload files to the project "Downloads" tab.' Another message says: 'For Git repositories, you can add the following line to your .netrc file recording your password for each repository domain you have commit access to. Make sure you have set the permissions on .netrc so that only your user account can read it.' At the bottom, the netrc entry is shown: 'machine code.google.com login khso192@gmail.com password ea7Dm9kR8fQ8'.

*Appendix - Subversion

❖ Subversion(SVN)

AnkSvn-2.5.12040

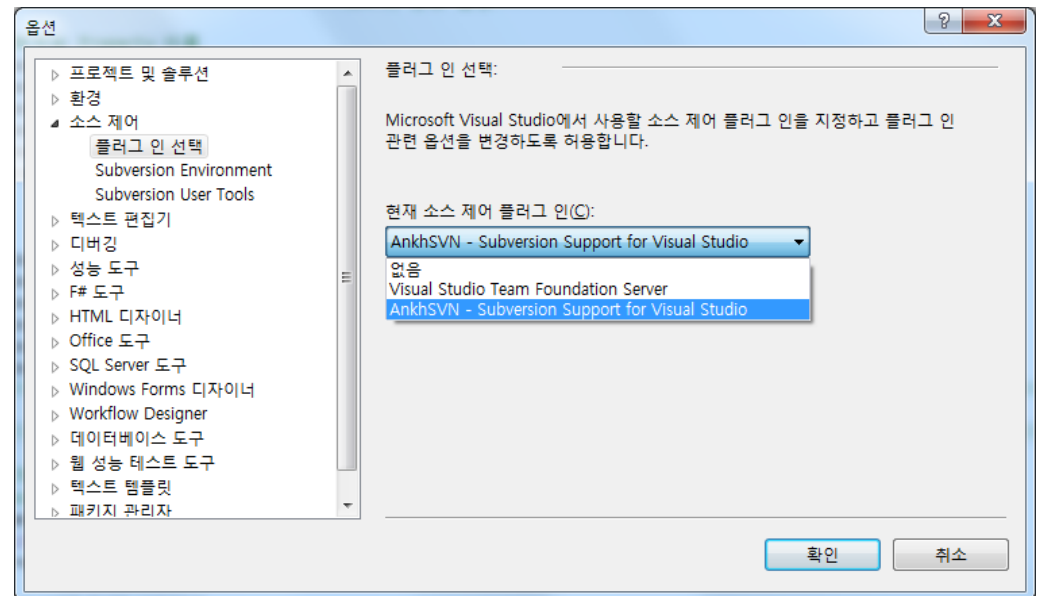
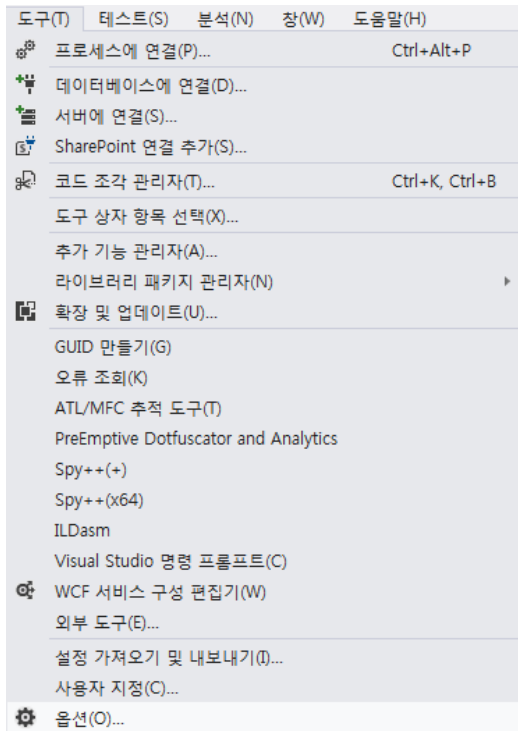


*Appendix - Subversion

❖ Subversion(SVN)

AnkhSvn-2.5.12040

도구 - 옵션 - 소스 제어 - 플러그인 - **AnkhSVN**으로 선택



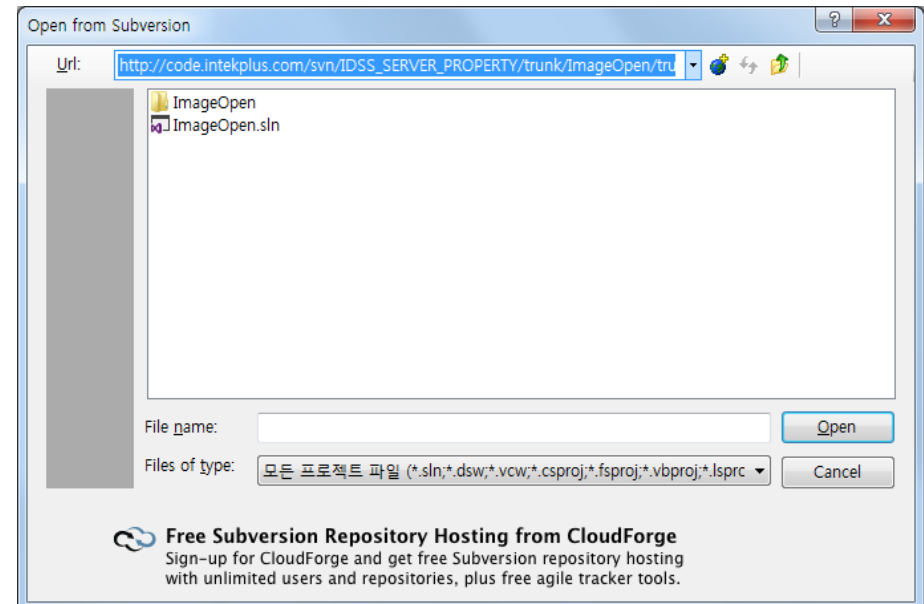
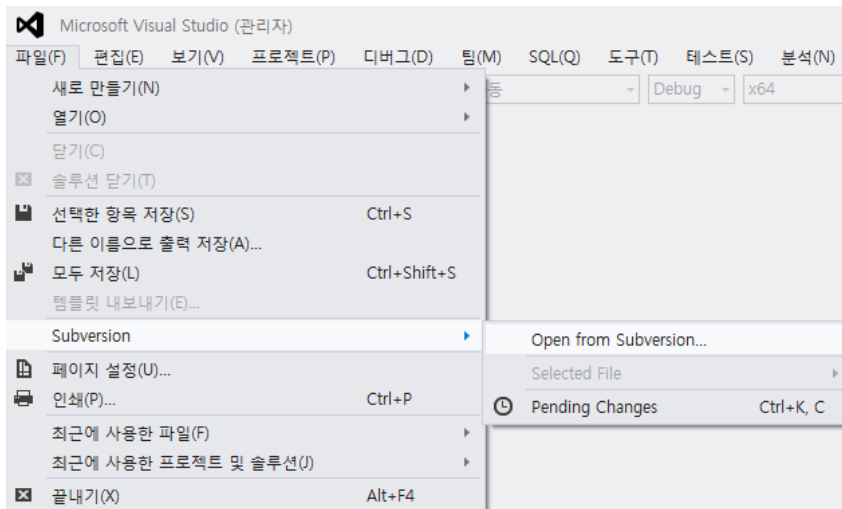
*Appendix - Subversion

❖ Subversion(SVN)

AnkhSvn-2.5.12040

파일 - Subversion - Open from Subversion - Url을 입력한다.

http://code.intekplus.com/svn/IDSS_SERVER_PROPERTY/trunk (<https://isl-test-svn.googlecode.com/svn/trunk/>)

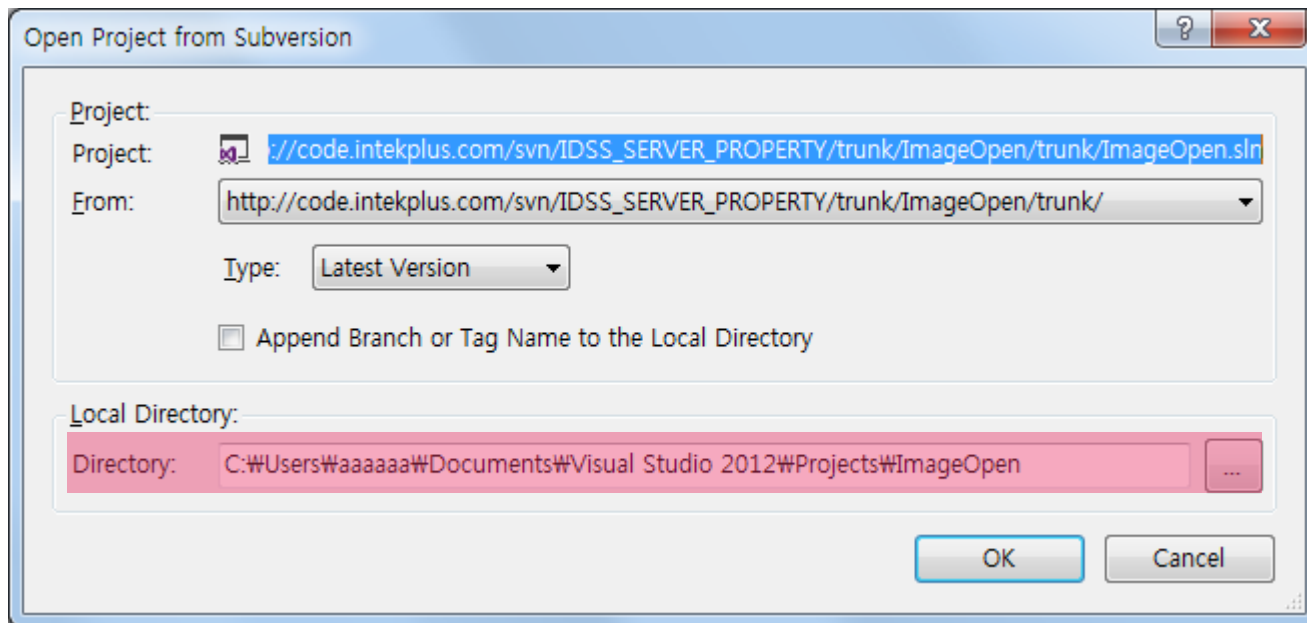


*Appendix - Subversion

❖ Subversion(SVN)

AnkhSvn-2.5.12040

원하는 솔루션을 선택하고, 솔루션이 저장될 디렉토리를 설정한다.



*Appendix - Subversion

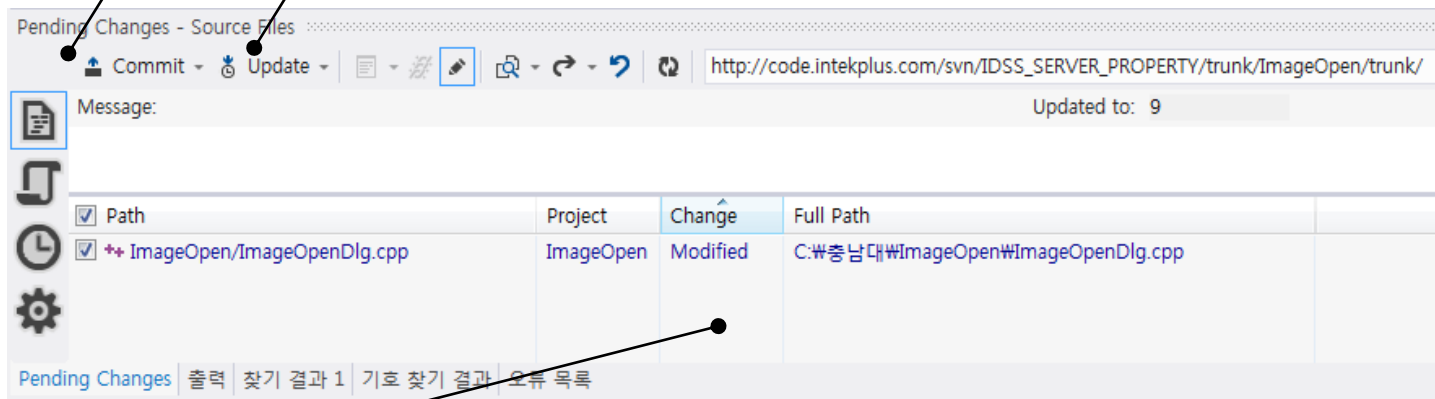
❖ Subversion(SVN)

Pending Changes

local 파일의 변경(수정, 추가, 삭제)을 repository에 저장한다.

이 때, 메시지 창에 메시지를 같이 입력하여, 다른 사람들도 어떠한 부분이 수정되었는지 알 수 있게 한다.

local의 파일을 repository와 비교하여 최신 버전의 상태로 갱신한다.



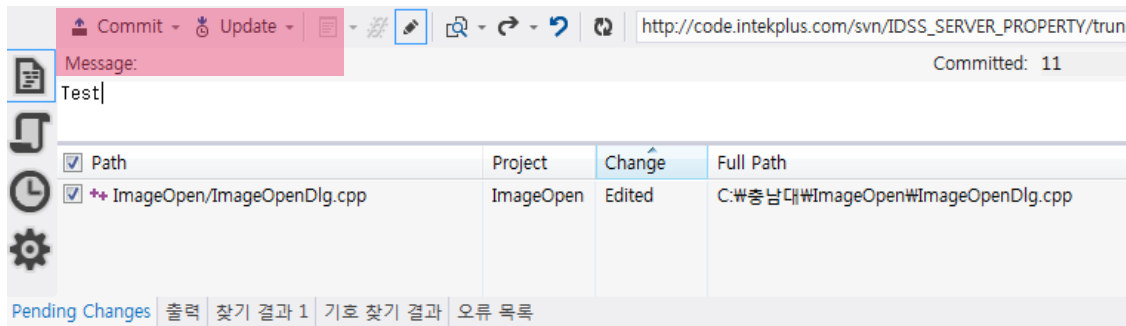
repository와 local에서 동시에 변경된 경우 Local 파일을 자동으로 Merge 하지만, Merge 실패 시 충돌(**Conflict**) 상태로 변경될 수 있다.

*Appendix - Subversion

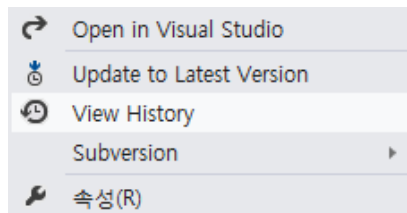
❖ Subversion(SVN)

Pending Changes 사용예

다음과 같이 메시지를 입력하고 Commit 버튼을 누르면, 메시지와 함께 local의 변경사항이 repository에 저장된다.



History View를 보면, 이전 과정에서 변경하였던 내용이 메시지와 함께 저장된 것을 확인할 수 있다.



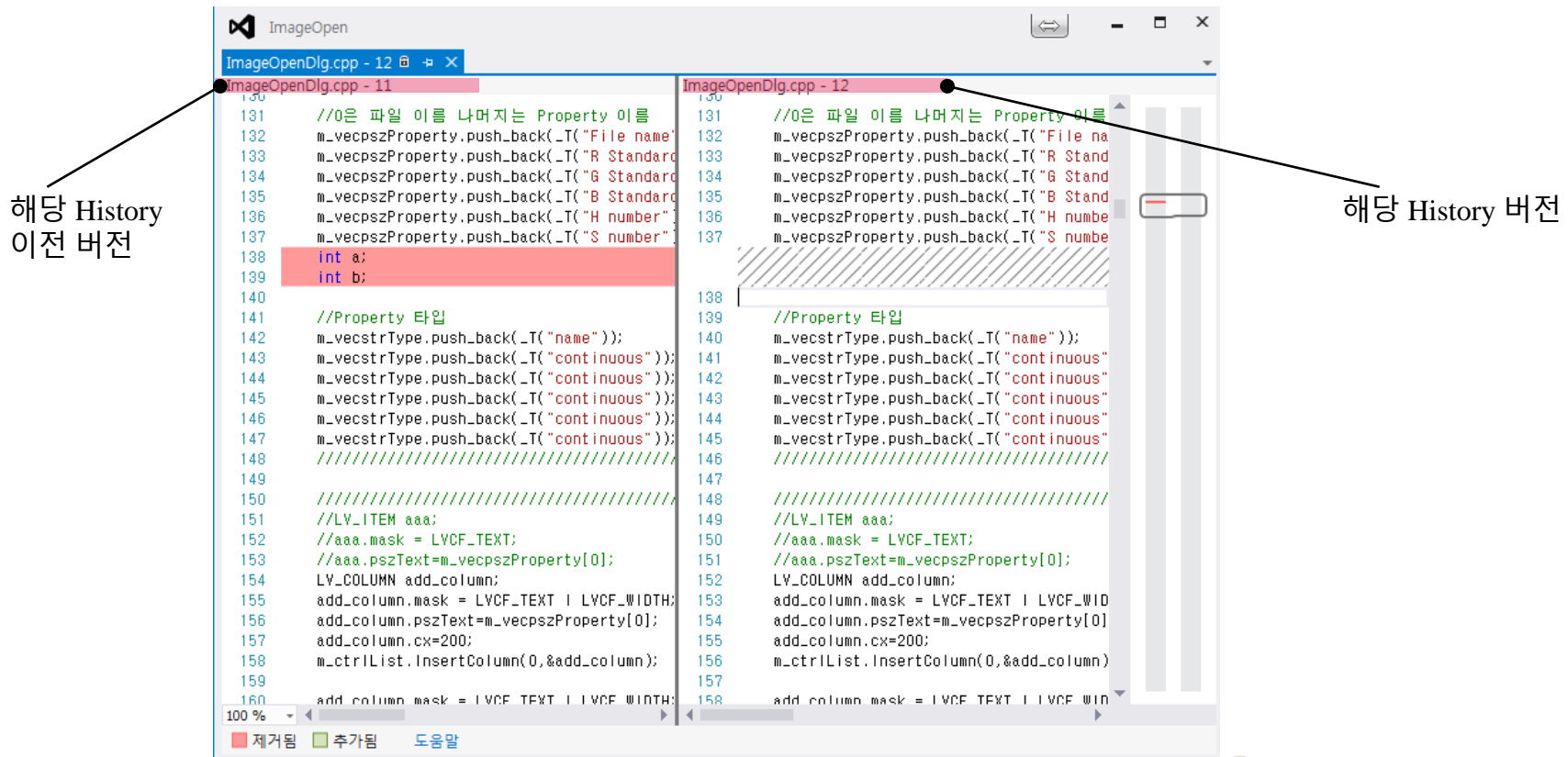
Revision	Author	Date	Message
12	cnu001	2015-01-06 오후...	Test
11	cnu001	2015-01-06 오후...	aaaa
10	09008	2015-01-06 오후...	멤버변수 사용 상관없는부분 정리
9	cnu001	2015-01-06 오후...	
8	cnu001	2015-01-06 오후...	수정후
7	cnu001	2015-01-06 오전...	수정전
6	cnu001	2015-01-06 오전...	
5	cnu001	2015-01-06 오전...	
4	cnu001	2015-01-06 오전...	수정점검
2	cnu001	2015-01-06 오전...	신규생성

*Appendix - Subversion

❖ Subversion(SVN)

Pending Changes 사용예

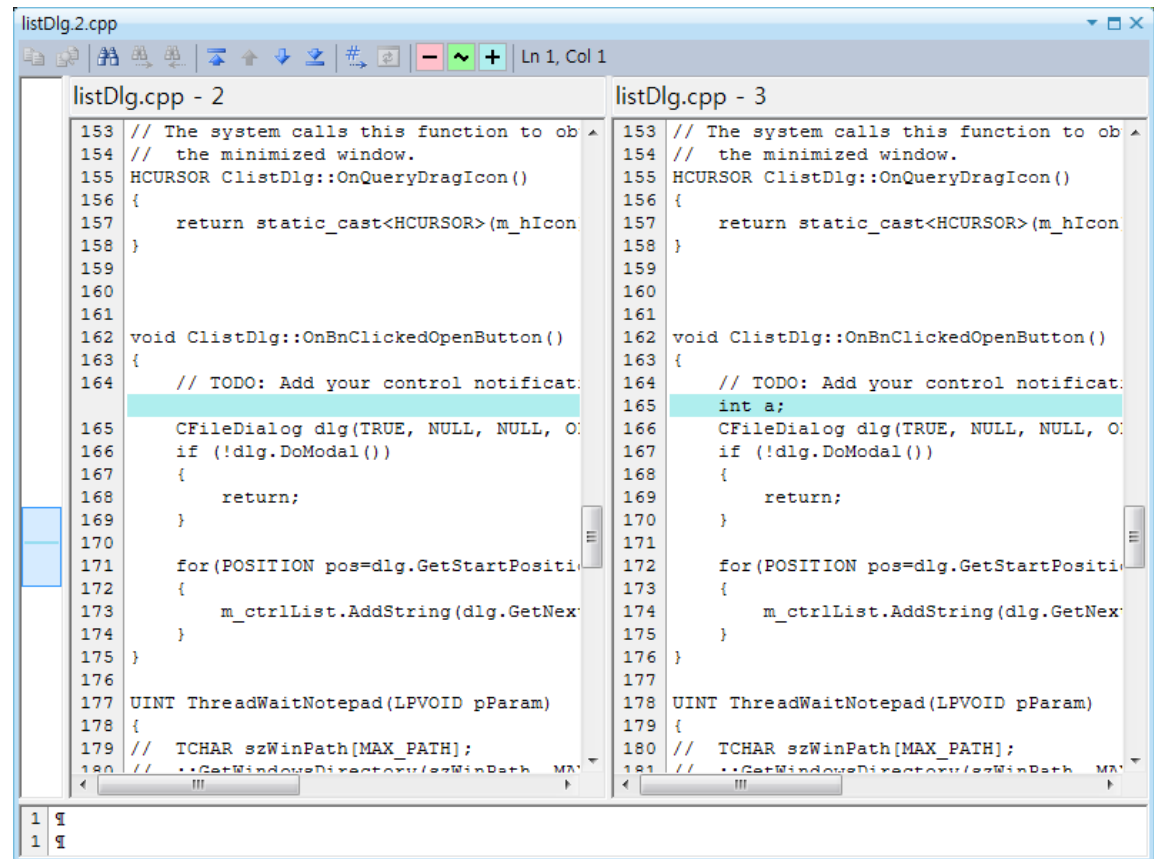
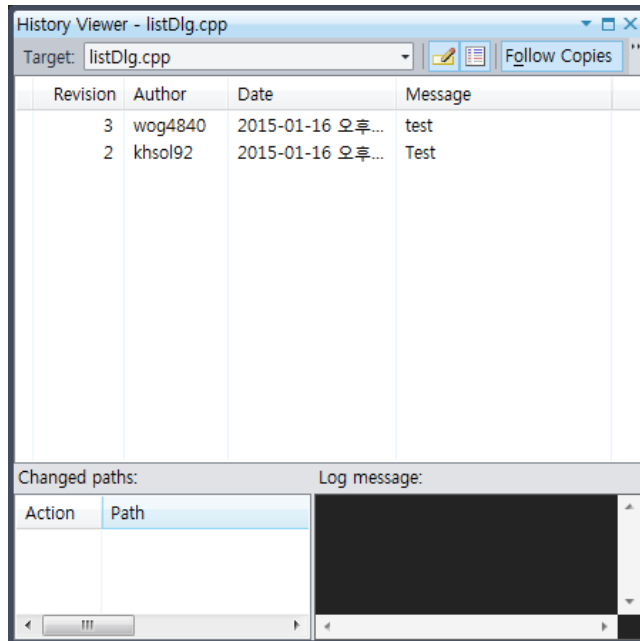
History View에서 History를 더블 클릭하면, 이전과 어떤 부분이 달라졌는지를 표시해준다.



*Appendix - Subversion

❖ Subversion(SVN)

Pending Changes 사용예



*Appendix – STL Vector

❖ STL의 구성

- Container : 객체들을 저장하는 객체 혹은 클래스
- Iterator : 컨테이너에 저장된 요소를 순회하고 접근하는 객체 혹은 클래스
- Algorithm : 데이터를 다루기 위한 일련의 함수
- Functor : 함수처럼 동작하는 객체로 operator 연산자를 오버로딩 한 클래스의 객체

*Appendix – STL Vector

❖ Container

■ 시퀀스 컨테이너

- 자료의 선형적인 집합이다.
- 입력되는 자료에 대한 특별한 제약이나 관리 규칙이 없다.
- 사용자가 시퀀스의 임의 위치에 원하는 요소를 삽입, 삭제할 수 있다.
- 컨테이너 양 끝과 중간에서 발생하는 작업의 비용이 많이 차이난다.

eg. vector, list, deque

■ 연관 컨테이너

- 자료를 일정한 규칙에 따라 조직화 하여 관리한다.
- 요소가 추가되는 위치를 결정할 수 없다.
- 정렬이나 해시 등의 방법을 통해 삽입되는 자료를 항상 일정한 기준에 맞는 위치에 저장해 검색속도가 빠르다.
- 요소의 추가, 삭제, 검색의 비용이 어디서나 동일하다.

eg. vector, list, deque

*Appendix – STL Vector

❖ Iterator

■ 특징

- 컨테이너 내부의 객체를 가리키고 접근할 수 있어야 한다.
- 컨테이너 내부의 모든 객체를 순회할 수 있어야 한다.

■ 반복자 기능에 따른 종류

- 입력 반복자(input iterator) : 현재 위치의 데이터를 읽기만 가능한 반복자
- 출력 반복자 (output iterator) : 현재 위치의 데이터를 쓰기만 가능한 반복자
- 순방향 반복자 (forward iterator) : 입력, 출력 반복자 기능 + 순방향으로 이동 가능한 반복자
- 양방향 반복자(bidirectional iterator) : 순방향 반복자 기능 + 역방향으로 이동 가능한 반복자
- 임의 접근 반복자(random access iterator) : 양방향 반복자 기능 + 반복자의 산술 연산이 가능한 반복자

*Appendix – STL Vector

❖ STL Vector의 특징

- 동적 배열 구조를 C++로 구현한 것이다.
- 템플릿 클래스로 어떤 타입이라도 저장 가능하다.
- 요소에 접근하거나, 앞이나 뒤에 요소를 추가, 삭제가 가능하고, 크기를 알 수 있는 멤버 함수를 제공한다.
- 연속 메모리 기반 컨테이너로 메모리가 자라나면 메모리를 재할당한다.
- 임의 접근 반복자로 산술연산이 가능하다.
(*모든 연속메모리 기반 컨테이너는 임의 접근 반복자를 제공)
- 읽기만 가능한 반복자를 사용한다면, `const_iterator`를 사용한다.

*Appendix – STL Vector

❖ STL Vector의 특징

- <Vector>헤더가 필요하다.
- 선언은 다음과 같은 형식으로 이뤄진다.

```
std::Vector<type> name;
```

- 데이터 삽입은 push_back(value)을 이용하여 이뤄진다.
- 최상위 데이터를 삭제하고자 할 때는 pop_back()을 이용한다.
- 원하는 데이터를 삭제할 때는 erase(iterator)을 이용한다.

*Appendix – STL Vector

❖ STL Vector의 주요 멤버

멤버	역할
assign	특정 원소로 채운다
at	특정 위치의 원소의 참조를 반환
back	마지막 원소의 참조를 반환
begin	첫 번째 원소의 랜덤 접근 반복자를 반환
clear	모든 원소를 삭제
empty	아무것도 없으면 true
end	마지막 원소 다음의(미사용영역) 반복자를 반환
erase	특정 위치의 원소나 지정 범위의 원소를 삭제
front	첫 번째 원소의 참조를 반환
insert	특정 위치에 원소 삽입
pop_back	마지막 원소를 삭제
push_back	마지막에 원소를 추가

*Appendix – STL Vector

❖ STL Vector의 주요 멤버

멤버	역할
rbegin	역방향으로 첫 번째 원소의 반복자를 반환
rend	역방향으로 마지막 원소 다음의 반복자를 반환
reserve	지정된 크기의 저장 공간을 확보
size	원소의 개수를 반환
swap	두 개의 vector의 원소를 서로 맞바꾼다.

*Appendix – STL Vector

❖ STL Vector 예제 : Vector의 기능을 활용한 예제로 값을 저장하고, 삭제하는 예제를 만들어 보았다.

```
#include <vector>
#include <iostream>

using namespace std;
int main ()
{
    vector <int> number;

    cout<<"push_back"<<endl;
    for(int i=0; i<10; i++)
    {
        number.push_back(i);          //값 저장
    }
    vector <int>::iterator itr = number.begin();

    for(int i=0; i<number.size(); i++)
    {
        cout<<number[i]<<endl;
    }

    cout<<endl<<"size : "<<number.size()<<endl;

    cout<<endl<<"pop_back"<<endl;
    number.pop_back();                //마지막에 있는 값(10)을 삭제
    for(int i=0; i<number.size(); i++)
    {
        cout<<number[i]<<endl;
    }

    cout<<endl<<"erase"<<endl;
    while(itr != number.end())
    {
        if(*itr == 5)
        {
            itr=number.erase(itr);    //5에 해당하는 값을 삭제
        }
        else itr++;
    }

    for(int i=0; i<number.size(); i++)
    {
        cout<<number[i]<<endl;
    }
}
```



```
push_back
0
1
2
3
4
5
6
7
8
9
```

size : 10

```
pop_back
0
1
2
3
4
5
6
7
8
```

```
erase
0
1
2
3
4
6
7
8
```

계속하려면 아무 키나 누르십시오 . . .

Q & A