Check new file uploaded on BB and start doing this Lab.

CSCW-415: Software Quality and Metrics

Labs

Labs 5 and 6: Code Size (Lines of Code: LOC)

For these labs we will be using a tool; “SourceMonitor”
https://www.campwoodsw.com/sourcemonitor.html

Students need to install, understand, and read help of “SourceMonitor”.

Students need to give input, to SourceMonitor, many java, c/c++, and html code files and **analyze** the results. Students need to measure/analyze these source code files by changing multiple parameters; for example; blank lines, comments, comments on multiple lines, keywords, new methods (many), decision statements (if/else) loops, etc. Furthermore, they need to compare the results achieved through this tool with the results they will manually calculate (inspection). They also need to write a **summary** of what they have analyzed/measured.

Students need to answer the following questions in their short reports:

1- Write what measurements/tests you are going to perform? (e.g. code size etc. see other types of measurements you can perform, in the help section of SourceMonitor)
2- Which programming language source code files you have tested?
3- How many files you have tested? (for each programming language)
4- What changes (e.g. changing parameters as described above) you made in each file after first test?
5- What was the result of tests after changes?
6- Presentation of results in tabular and graphical formats.

Notes:

1- Students are required to work in groups.
2- Students can use their existing programs also (which they have already developed in other classes as assignments)

Introduction:

SourceMonitor is a tool that measures and records source code metrics (i.e. to know what is inside a body of code). It also preserves these counts and provides overviews of a project's history. SourceMonitor is a tool for coders and their supervisors. It is intended for use by programmers as they write code, to expose to them information about the code they are creating. Given this information, they have an opportunity to improve the way they work, the tools they use and ultimately the quality of the code they write. SourceMonitor is a tool for programmers who work in the 32 bit or 64 bit Windows world; Windows XP, Windows Vista, or Windows 7. SourceMonitor can parse only plain ASCII text files. The goal of SourceMonitor is to provide you with a quick overview of the **size and complexity of your code**. (Source: Help Topics in SourceMonitor)

Step 1:

Download it from here:
https://www.campwoodsw.com/SM/SMSetupV3516.exe

Step 2:

Install it and start the SourceMonitor.

Step 3:

Create a folder and add different .java (or c/c++/html/etc.) files there. These files would be used for measuring/testing code size etc.

Step 4:

Create Project as described below.

Step 5:

Create a new checkpoint as described below. (The dialog box in this step will help you to select directory in which you have added .java (or c/c++/html/etc.) files).

After clicking OK button, you will see the project window and different results.

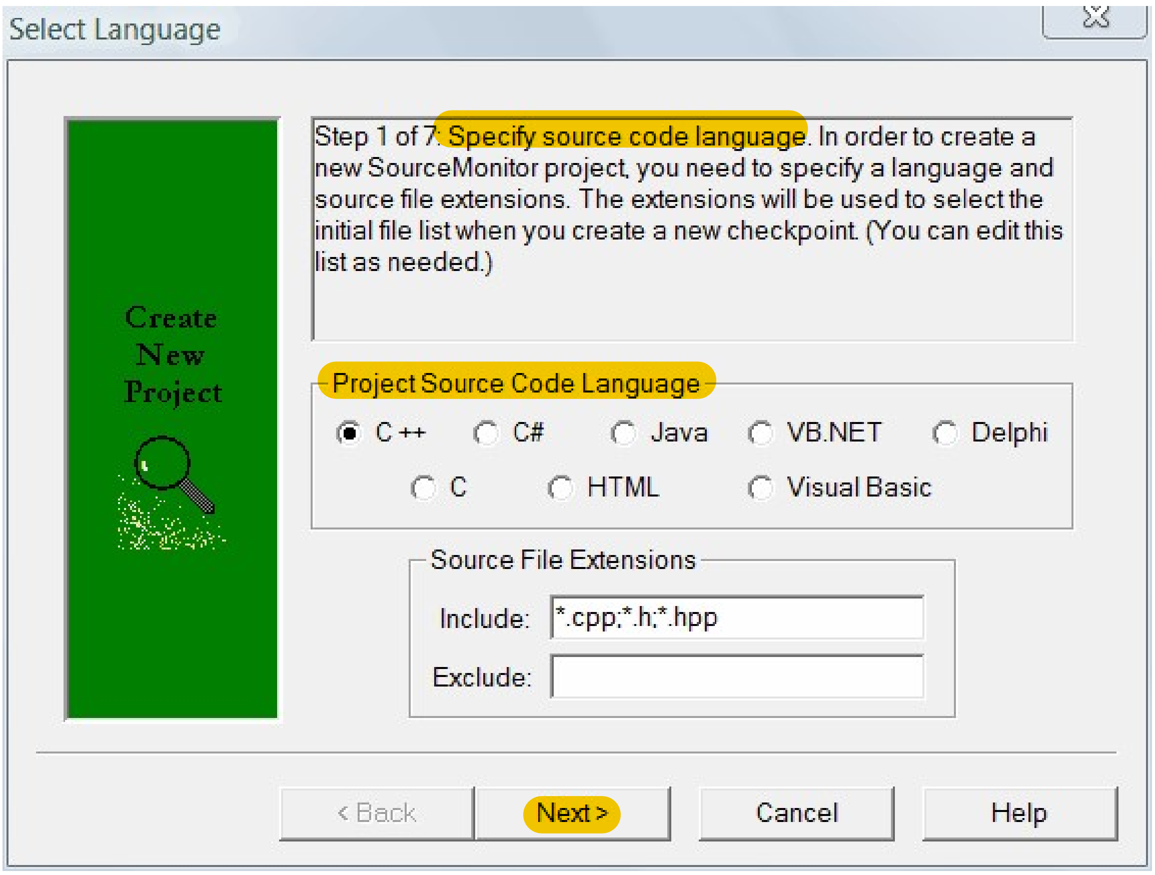
Note: Save results in screenshots so that it can be added in report. (for example like given below). By right clicking in “project view” window you can use extra menu options for more results.

How to Create a Project

Top Previous Next

You create a new project in the create project wizard where you specify a language, specify one or more source code directories, and give the project a name. (You can also create a project in a command line script.)

There are three ways to launch the new project wizard: 1) click on the **File | New Project** menu item, 2) click on the new project button in the tool bar, or 3) hold down the Ctrl shift button and tap the "N" key. The first page of the create project wizard appears as shown below.



Step 1 of 7: Specify source code language. In order to create a new SourceMonitor project, you need to specify a language and source file extensions. The extensions will be used to select the initial file list when you create a new checkpoint. (You can edit this list as needed.)

Project Source Code Language: [✓ C++] C# Java VB.NET Delphi C HTML Visual Basic

Source File Extensions

Include: *.cpp;*.h;*.hpp

Exclude:

On this page all you need to do is select the language used in your source code files. (If you have more than one language in your project, you must create a different SourceMonitor project for each language.) When you select a language, you will see a default list of file extensions for this language as well as a list of extensions to be excluded. (You can modify these extension lists on the Options dialog tab for each language.) If your files use different extensions, edit the list as desired. These extension are used to create a suggested list of files for the first checkpoint in your new project. You can modify this list before the checkpoint is created.

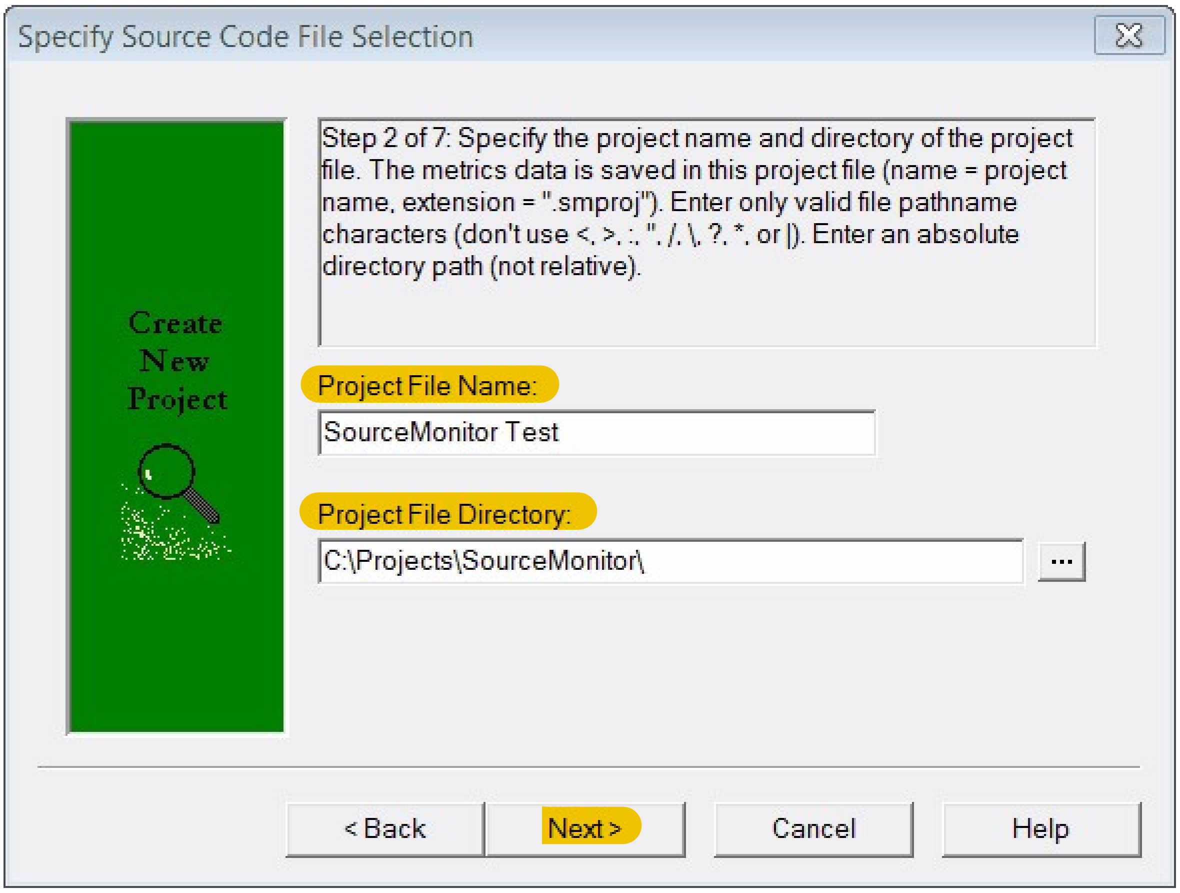
Click on the **Next** button to advance to Step 2.

Create a Project - Step 2

In step 2 (see below) you will specify the location of your project file. This file holds the history of your project as well as your project's options. You must specify a name for your project, and this name must be a valid file name (SourceMonitor will add extension ".smproj" for form the actual file name). That means you can not use any of the following characters in your project name:

< > : " / \ ? * |

You must also specify a directory where your new project file will be saved. You can click on the "**...**" button and browse to a directory. However, be careful to select a directory to which you have write access.



Step 2 of 7: Specify the project name and directory of the project file. The metrics data is saved in this project file (name = project name, extension = ".smproj"). Enter only valid file pathname characters (don't use <, >, :, ", /, \, ?, *, or |). Enter an absolute directory path (not relative).

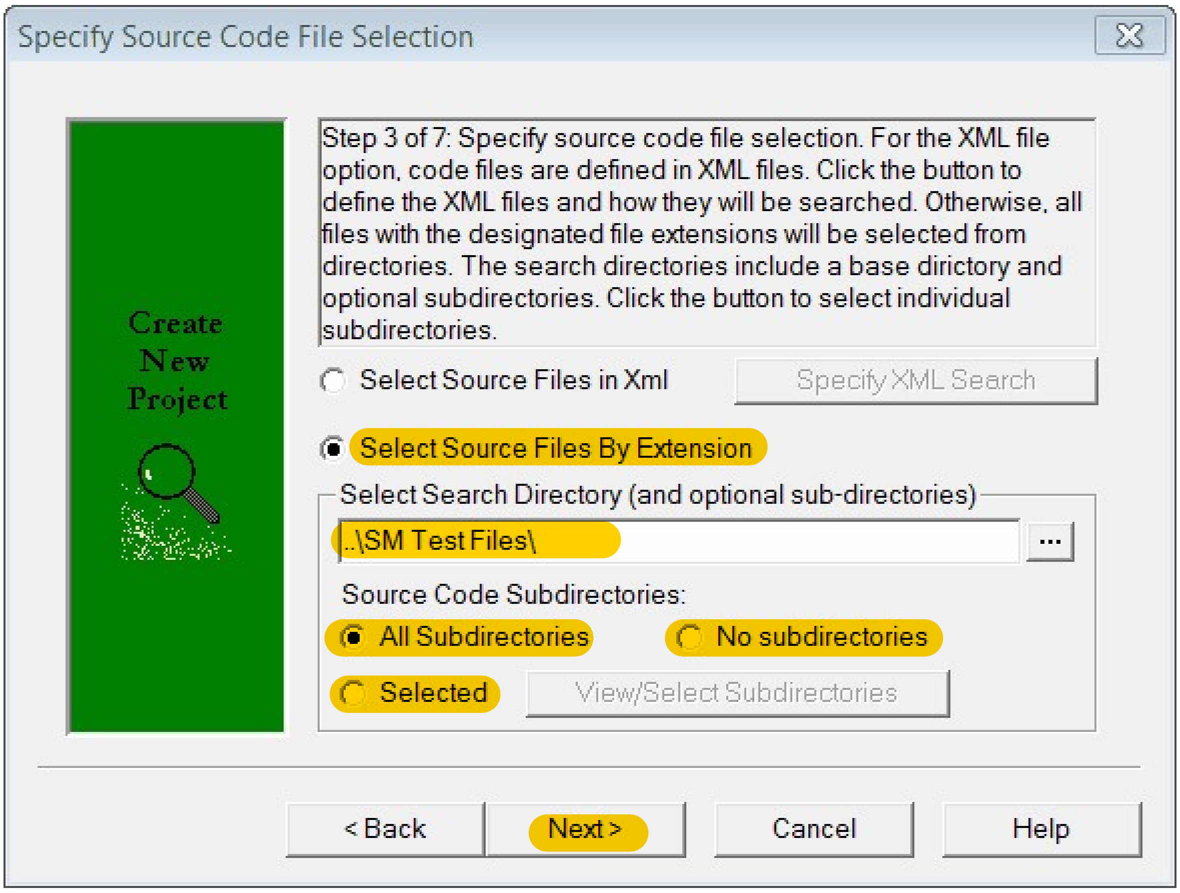
Project File Name: SourceMonitor Test

Project File Directory: C:\Projects\SourceMonitor\

Once again, you click on the Next button to move to the next step, Step 3.

Create a Project - Step 3

This step asks you to specify where your source code is located. The paths to your source code files are stored in the project file. If you specify a relative path, you can move your project file and nested source code files together whenever you wish. However, if you specify an absolute source code path, you must modify your project's properties if you ever move your project file to a different directory.



Specify Source Code File Selection

Step 3 of 7: Specify source code file selection. For the XML file option, code files are defined in XML files. Click the button to define the XML files and how they will be searched. Otherwise, all files with the designated file extensions will be selected from directories. The search directories include a base dirictory and optional subdirectories. Click the button to select individual subdirectories.

Select Source Files in Xml [✓ Select Source Files By Extension]

Select Search Directory (and optional sub-directories): ..\SM Test Files\

Source Code Subdirectories: [✓ All Subdirectories] No subdirectories Selected

This step asks you to specify how files will be selected for the checkpoints in your project. You have two options: 1) specify files in one or more XML files, or 2) specify files by file name extension and one or more search directories.

Option 1:

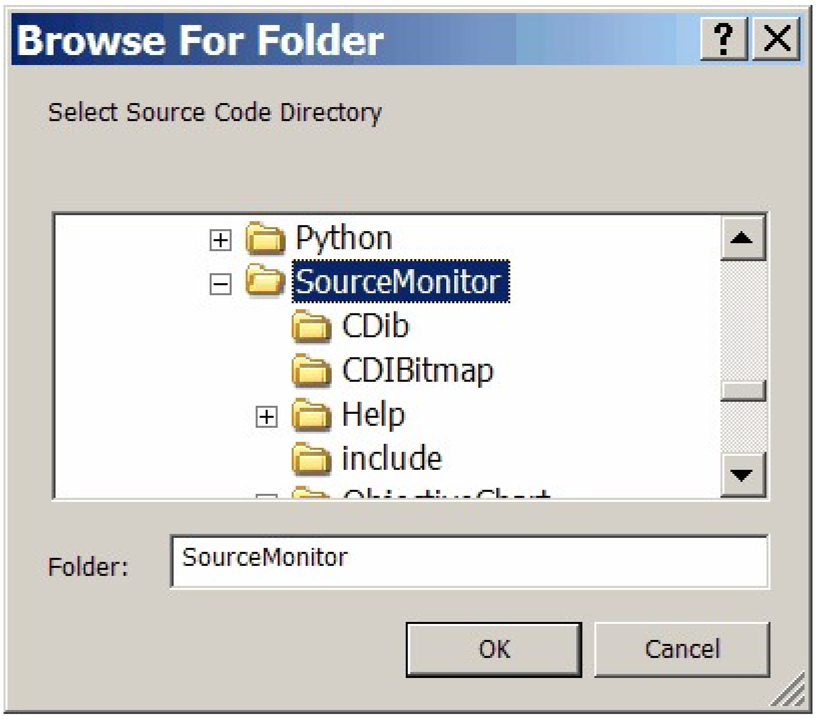
When you select the top radio button, the **Specify XML Search** button will be enabled. This option enables SourceMonitor to select the files to be included in a checkpoint from one or more XML files. (Often your development IDE tool will maintain a list of the source files in a project in an XML file.) You will need to specify the XML file(s) that list the source code files, as well as the base directories to be used in resolving relative file pathnames.

When you click on the **Specify XML Search** button, you will see the extraction specification dialog. You can specify source code file extraction parameters in this dialog.

When you have completed your extraction specification, click on the **Next** button once more to display step 4.

Option 2:

If you prefer to select files by file name extension, select the lower radio button. For the source code file directory field, you can click on the "**...**" button to the right of the field and browse to select a directory. When you have the correct directory just click on the **OK** button as shown below:



If your source code is located in subdirectories under the directory you selected, you can either click on the **Selected** option and click on the **View/Select Subdirectories** button to display a dialog where you can pick the subdirectories that you want to include in the project. Alternatively, you can select the **All Subdirectories** option and SourceMonitor will include in your project the source code files in all of the subdirectories below the source directory.

Note that you can specify a relative search directory. For example, if your project file is in directory

C:\Projects\Campwood

then you can replace the source code search directory shown above with

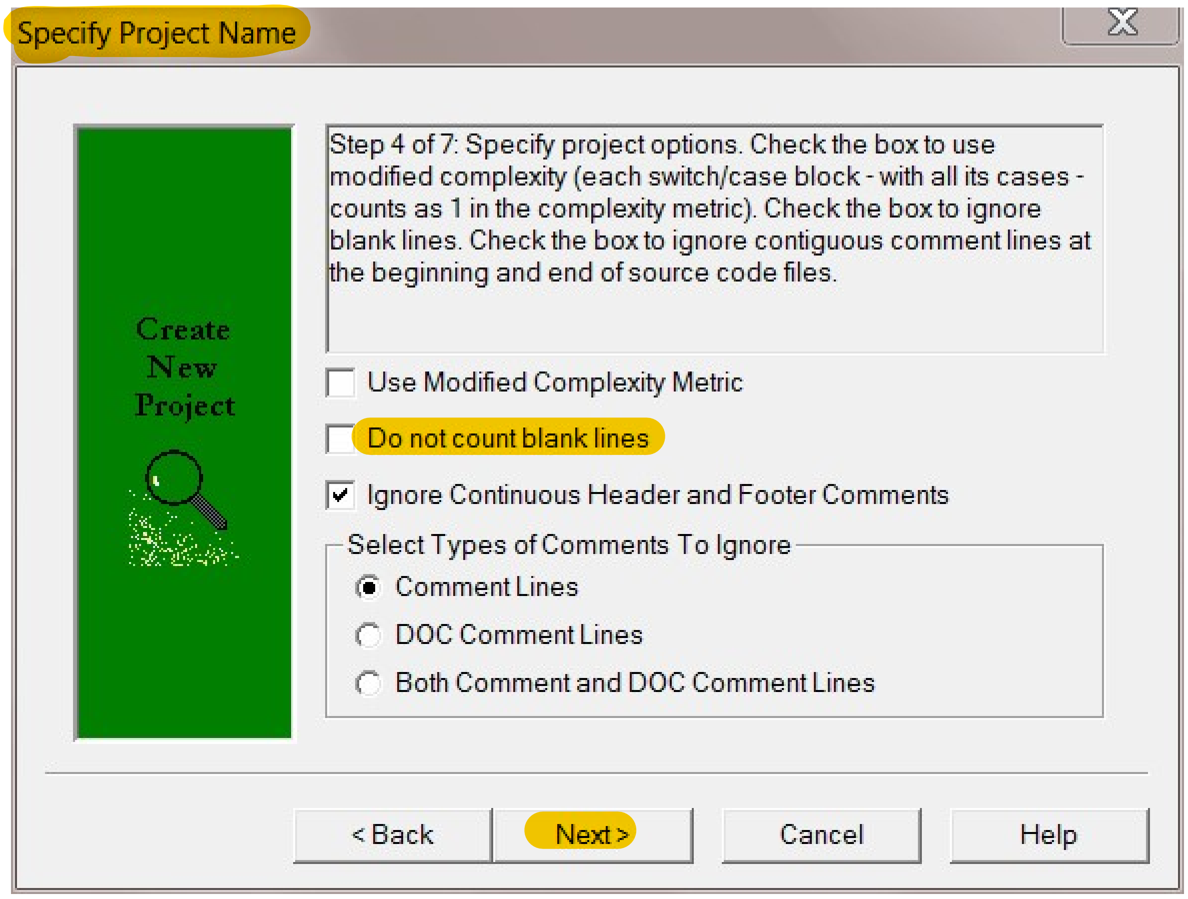
..\SM Test Files

to tell SourceMonitor to look in C:\Projects\SM Test Files for source code files.

Click on the **Next** button once more to display Step 4.

Create a Project - Step 4

This step is simple: just check the option boxes you wish to select for your project. You can change these options later in the Project Properties dialog.



Step 4 of 7: Specify project options. Check the box to use modified complexity (each switch/case block - with all its cases - counts as 1 in the complexity metric). Check the box to ignore blank lines. Check the box to ignore contiguous comment lines at the beginning and end of source code files.

☐ Use Modified Complexity Metric

☐ Do not count blank lines

☑ Ignore Continuous Header and Footer Comments

Select Types of Comments To Ignore

- ● Comment Lines
- ○ DOC Comment Lines
- ○ Both Comment and DOC Comment Lines

The default complexity metric is computed by counting the cases within each switch statement. With the Modified Complexity option enabled, each switch block is counted just once (the individual case statements do not contribute to the Modified Complexity).

SourceMonitor counts all physical lines in the Lines metric. If you don't want blank source code lines counted, select the **Do not count blank lines** check box.

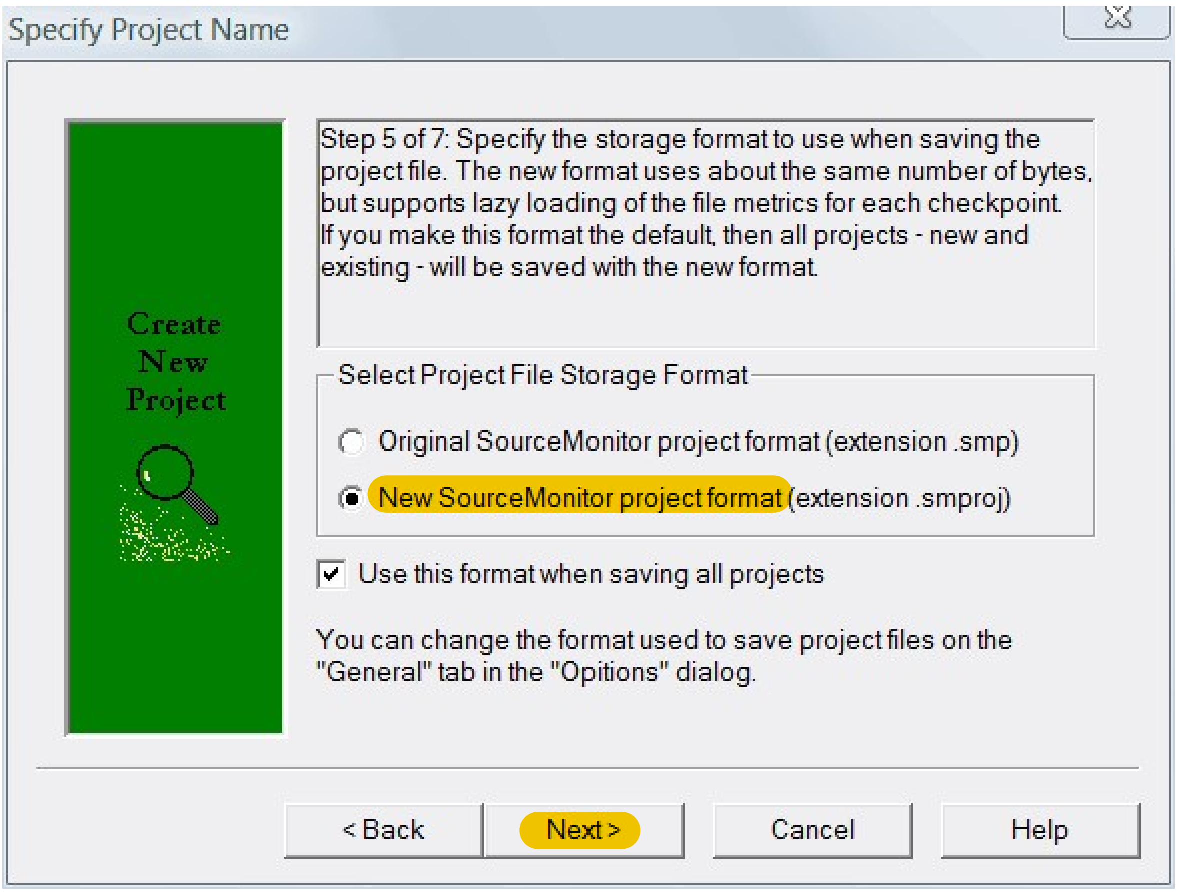
Some developers like to put headers and/or footers in every source code file. These are contiguous comment lines that contain such information as the component to which the code belongs, the author's name, or a file history automatically generated by a version control utility such as CVS or Visual SourceSafe. Since these sections do not represent code comments, you may want SourceMonitor to ignore your headers and footers. If so, select the **Ignore Continuous Header and Footer Comments** check box.

For some languages, SourceMonitor recognizes DOC comments in addition to normal comments. In these cases, you can select which comments you want SourceMonitor to ignore.

When the above options are specified, click on the **Next** button to move on to step 5.

Create a Project - Step 5

Step 5 gives you the option to continue to use the pre-V3.0 project file format. You should use the new project file format introduced in V3.0 to enjoy fast loading of your project as it accumulates more and more metrics data.



Step 5 of 7: Specify the storage format to use when saving the project file. The new format uses about the same number of bytes, but supports lazy loading of the file metrics for each checkpoint. If you make this format the default, then all projects - new and existing - will be saved with the new format.

Select Project File Storage Format

○ Original SourceMonitor project format (extension .smp)

● New SourceMonitor project format (extension .smproj)

✓ Use this format when saving all projects

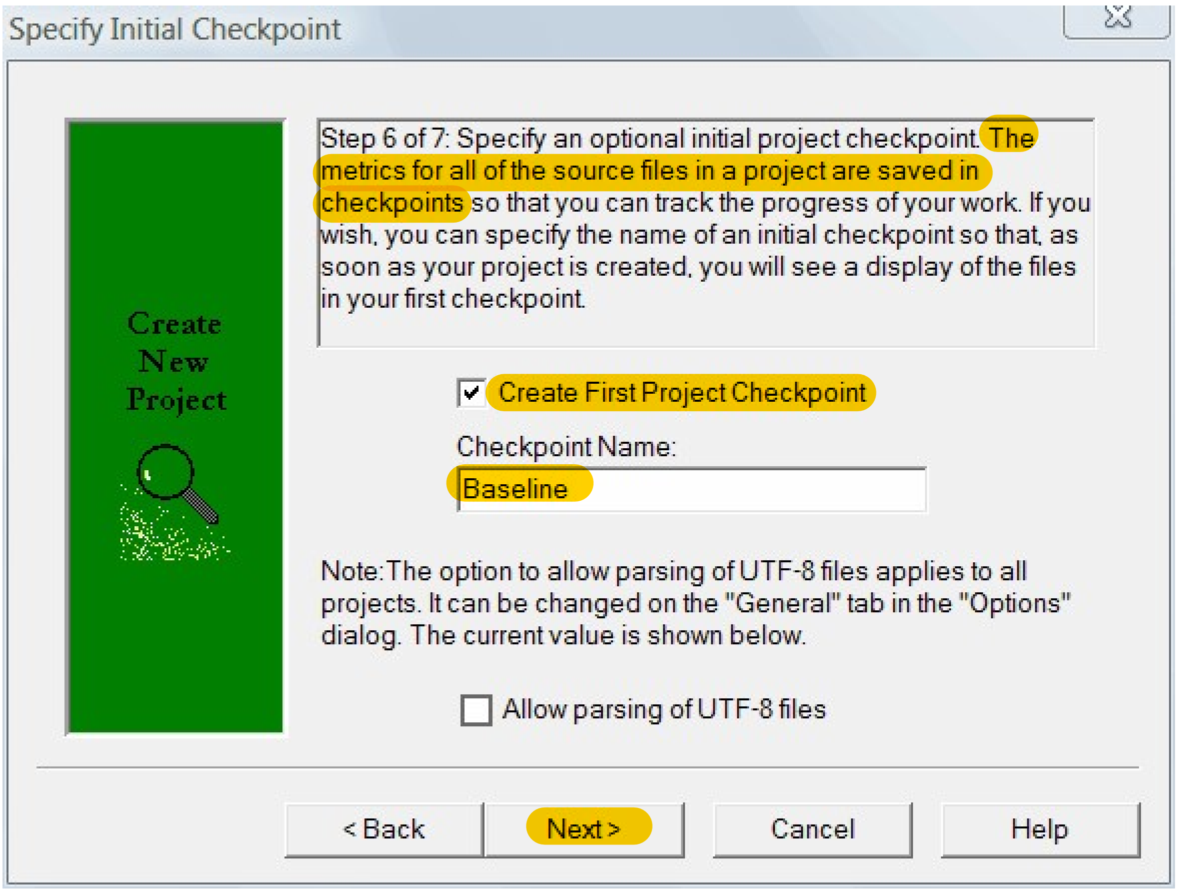
You can change the format used to save project files on the "General" tab in the "Opitions" dialog.

Click the Next button to advance to step 6.

Create a Project - Step 6

Top Previous Next

In step 6 you are given the option to create an initial checkpoint in your new project. The default settings are shown in the screen shot, below.



Step 6 of 7: Specify an optional initial project checkpoint. The metrics for all of the source files in a project are saved in checkpoints so that you can track the progress of your work. If you wish, you can specify the name of an initial checkpoint so that, as soon as your project is created, you will see a display of the files in your first checkpoint.

[✓] Create First Project Checkpoint

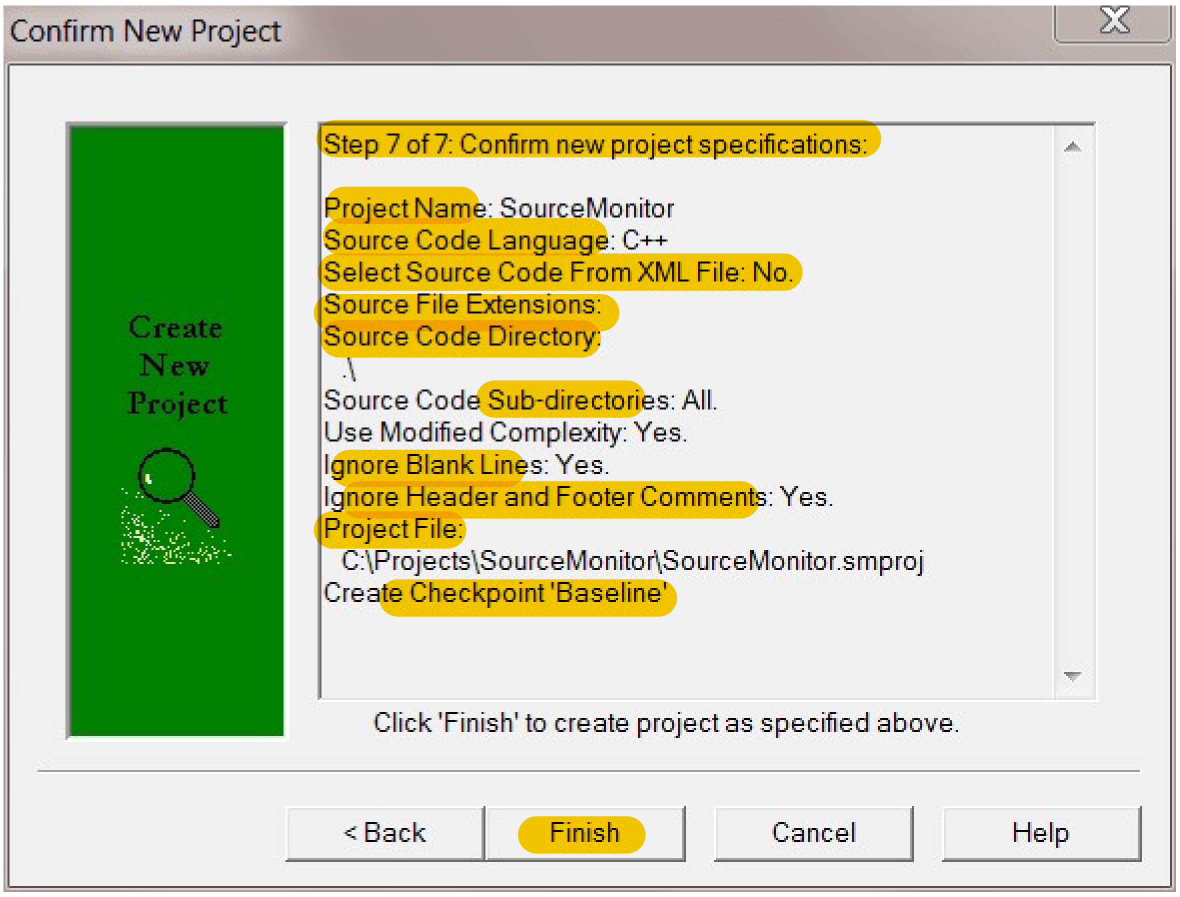
Checkpoint Name: Baseline

Note: The option to allow parsing of UTF-8 files applies to all projects. It can be changed on the "General" tab in the "Options" dialog. The current value is shown below.

[ ] Allow parsing of UTF-8 files

The UTF-8 file parsing option is important if your IDE saves your source code in files that begin with bytes that define a UTF format. If you select this option, SourceMonitor will attempt to parse files that begin with UTF-8 marker bytes. However, if such a file contains escape sequences for characters not in the basic ASCII character set then SourceMonitor will be unable to parse the file.

Click on the **Next** button one last time to examine and verify your options in the confirm page as shown below:



Step 7 of 7: Confirm new project specifications:

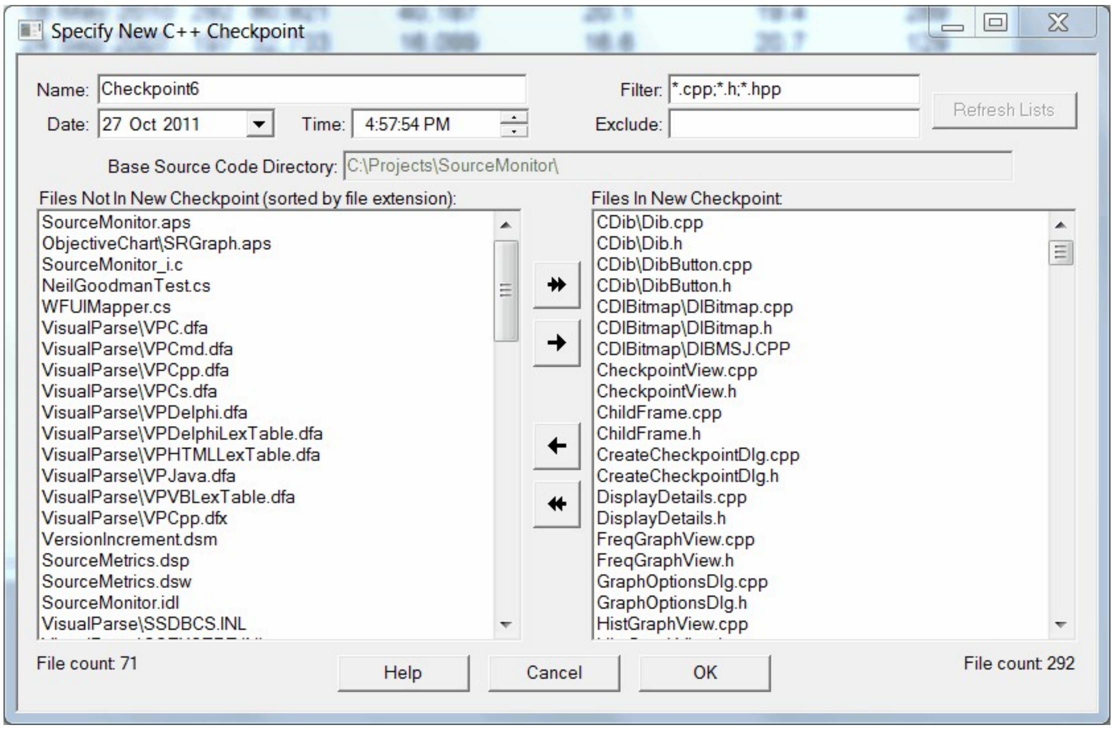
Project Name: SourceMonitor
Source Code Language: C++
Select Source Code From XML File: No.
Source File Extensions:
Source Code Directory:
 .\
Source Code Sub-directories: All.
Use Modified Complexity: Yes.
Ignore Blank Lines: Yes.
Ignore Header and Footer Comments: Yes.
Project File:
 C:\Projects\SourceMonitor\SourceMonitor.smproj
Create Checkpoint 'Baseline'

Click 'Finish' to create project as specified above.

That's all there is to it. If the new project data is correct, just click on the **Finish** button and you're done.

How to Create a New Checkpoint

When you are ready to add a new checkpoint to a project, open the Project View for the project. Then click on the New Checkpoint button on the tool bar, the **Checkpoint | New Checkpoint...** menu item or click on **New Checkpoint...** in the right click pop up menu. You will see the new checkpoint dialog, a sample of which is shown below:



Name: Checkpoint6
Filter: *.cpp;*.h;*.hpp
Date: 27 Oct 2011
Time: 4:57:54 PM
Exclude:
Base Source Code Directory: C:\Projects\SourceMonitor\

Files Not In New Checkpoint (sorted by file extension):

```
SourceMonitor.aps
ObjectiveChart\SRGraph.aps
SourceMonitor_i.c
NeilGoodmanTest.cs
WFUIMapper.cs
VisualParse\VPC.dfa
VisualParse\VPCmd.dfa
VisualParse\VPCpp.dfa
VisualParse\VPCs.dfa
VisualParse\VPDelphi.dfa
VisualParse\VPDelphiLexTable.dfa
VisualParse\VPHTMLLexTable.dfa
VisualParse\VPJava.dfa
VisualParse\VPVBLexTable.dfa
VisualParse\VPCpp.dfx
VersionIncrement.dsm
SourceMetrics.dsp
SourceMetrics.dsw
SourceMonitor.idl
VisualParse\SSDBCS.INL
```

File count 71

Files In New Checkpoint

```
CDib\Dib.cpp
CDib\Dib.h
CDib\DibButton.cpp
CDib\DibButton.h
CDIBitmap\DIBitmap.cpp
CDIBitmap\DIBitmap.h
CDIBitmap\DIBMSJ.CPP
CheckpointView.cpp
CheckpointView.h
ChildFrame.cpp
ChildFrame.h
CreateCheckpointDlg.cpp
CreateCheckpointDlg.h
DisplayDetails.cpp
DisplayDetails.h
FreqGraphView.cpp
FreqGraphView.h
GraphOptionsDlg.cpp
GraphOptionsDlg.h
HistGraphView.cpp
```

File count 292

This dialog has default values for your new checkpoint's name and creation date and time. Default checkpoint names are created by appending the next sequential number to the word "Checkpoint" where the previous number comes from the names of existing checkpoints. You can replace this default name with something more meaningful if you like.

You can leave the default date and time for the new checkpoint, or change it to whatever is appropriate for your source code. You can click on the arrow to the right of the date display to select a new date in a calendar. Likewise you can highlight any segment of the time and adjust its value with the spin buttons.

The file extension filter is displayed at the upper right. If your project selects files by extension, this is the filter that is used to select files for each new checkpoint. This filter may be specified for each language (see Options). A list of file extensions to be excluded is also displayed. This is useful when files have multiple extensions (more than one dot character) and the final extension is wanted, but a sub-extension followed by an included extension should be excluded from the new checkpoint file list. For example, in a C# project developed with Visual Studio, you will probably have just one extension, "*.cs", in your filter list. To exclude files automatically generated by Visual Studio you could add the following to your Exclude list: "*.Designer.cs". You can specify an excluded extensions list for each language (see Options).

In addition, you can use the exclude filter to exclude a group of files that start with the same characters. For example, you can exclude files MyFirstFile.cs and MySecondFile.cs with the following entry in the exclude filter list: "My*File.cs". Note that only one wildcard '*' character may be included in any one exclude list entry.

Note that if your project selects files from one or more XML files, the include and exclude filters are still used; however, they are applied to the file names extracted from your XML file(s) instead of to all files in specified directories.

You can make a temporary change to either the filter or the exclude list, then click on the **Refresh Lists** button to get a new list of files selected for a new checkpoint.

All of the files are shown with their paths relative to the Project's default Base Source Code Directory. Your new Checkpoint will be created with this base directory (you can change the base directory in the Options dialog).

The files shown in the list on the right side will become members of the new checkpoint. Those files in the list on the left are in the project directory (and subdirectories if you selected that option for your project), but have file name extensions that are not present in the filter or that are specified in the Exclude extensions list. The list on the left will also contain all Unicode files found; however, you can include UTF-8 files in your checkpoints if you select "Allow parsing of UTF-8 files" in the Options dialog. Note that UTF-8 files will be parsed properly by SourceMonitor only if they contain no Unicode characters.

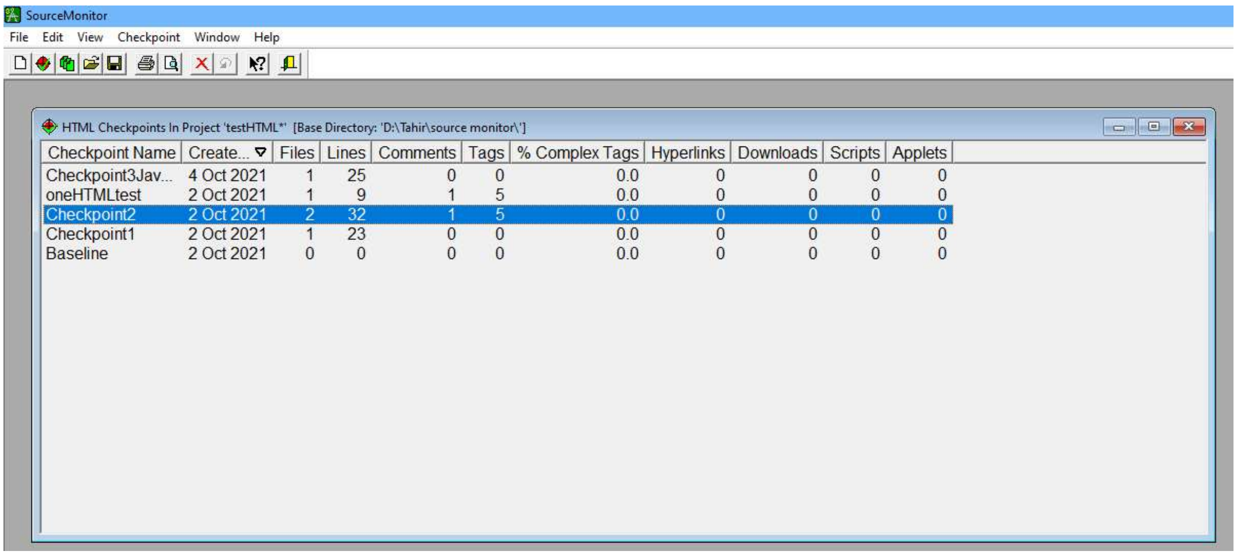
To move files from one side to the other, select them and then click the proper arrow button. Or double click on any file to move it to the other side. However, you will only be allowed to move plain ASCII files (and optionally UTF-8 files - see previous paragraph) into the checkpoint list on the right. Likewise empty files may not be added to a checkpoint.

This dialog can be moved or resized, and the new position and size will be used if you re-open this dialog later in your current SourceMonitor session.

When you click on the OK button, a progress dialog will appear. This dialog displays the name of each file as it is parsed. When the new checkpoint has been created, the time needed to create the checkpoint as well as the source lines processed per second are displayed in the status bar at the bottom of SourceMonitor's main window. (This display remains visible only until you ask SourceMonitor to perform another task.)

Some result screenshots:

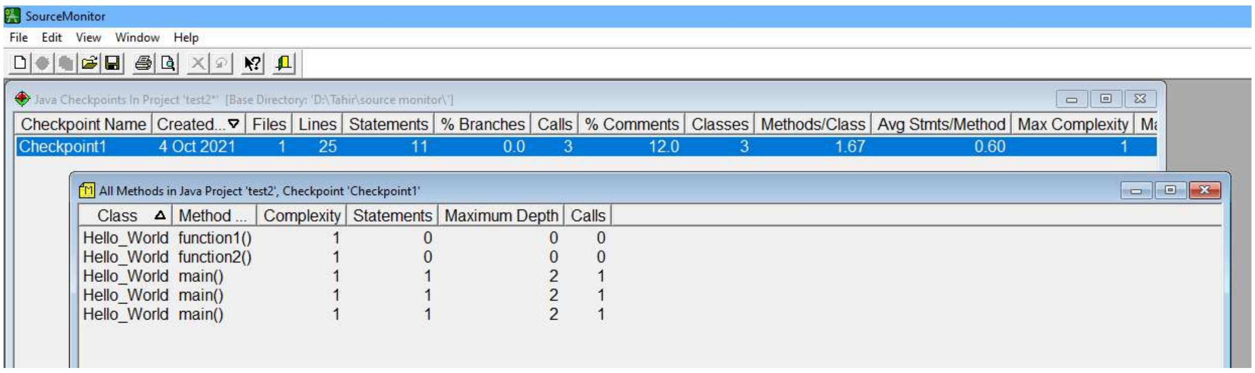
A view of project window:



HTML Checkpoints In Project 'testHTML*' [Base Directory: 'D:\Tahir\source monitor\']

Checkpoint Name	Create...	Files	Lines	Comments	Tags	% Complex Tags	Hyperlinks	Downloads	Scripts	Applets
Checkpoint3Jav...	4 Oct 2021	1	25	0	0	0.0	0	0	0	0
oneHTMLtest	2 Oct 2021	1	9	1	5	0.0	0	0	0	0
Checkpoint2	2 Oct 2021	2	32	1	5	0.0	0	0	0	0
Checkpoint1	2 Oct 2021	1	23	0	0	0.0	0	0	0	0
Baseline	2 Oct 2021	0	0	0	0	0.0	0	0	0	0

Try to get the results for different methods metrics: (like given below)



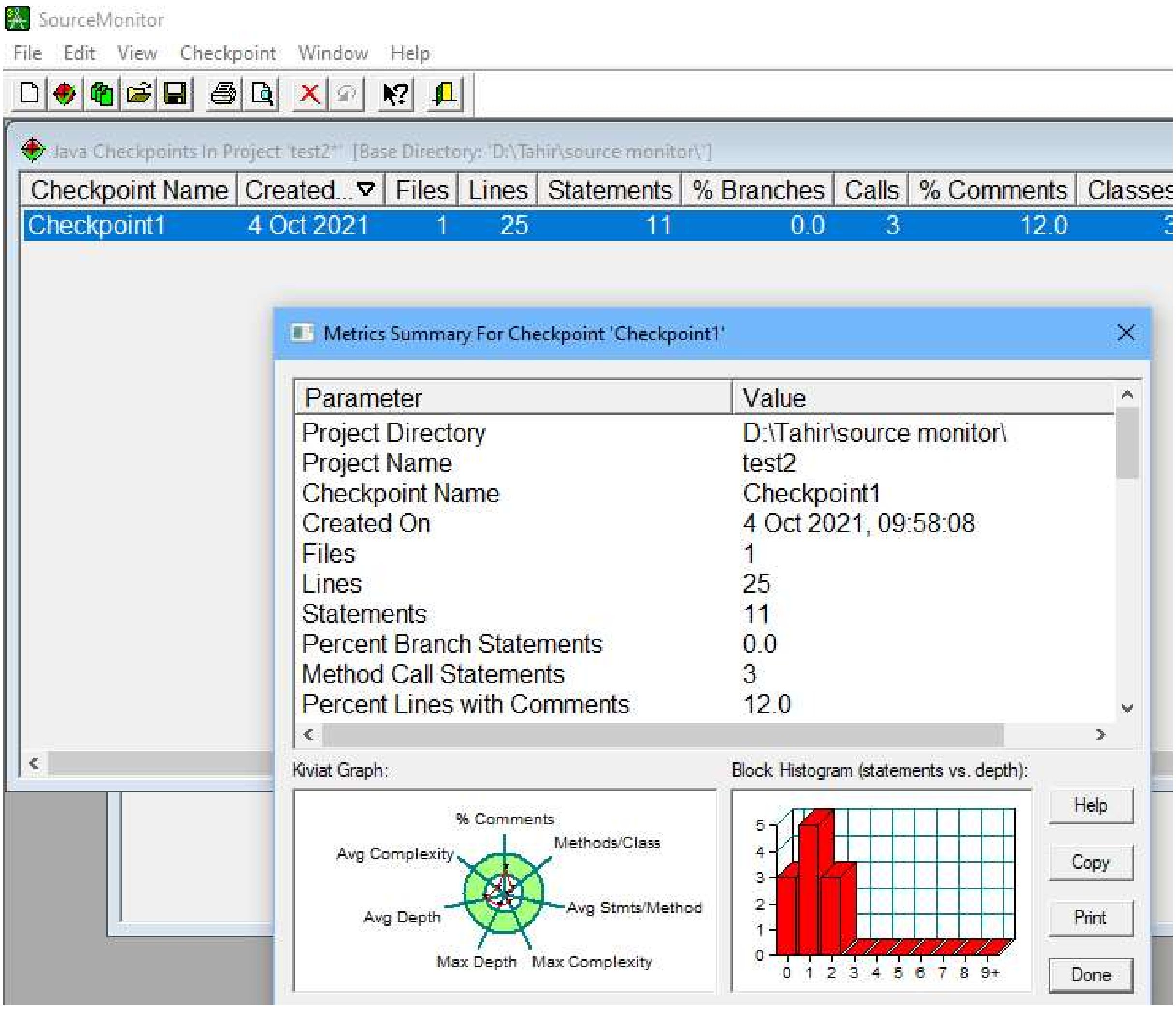
Java Checkpoints In Project 'test2*' [Base Directory: 'D:\Tahir\source monitor\']

Checkpoint Name	Created...	Files	Lines	Statements	% Branches	Calls	% Comments	Classes	Methods/Class	Avg Stmts/Method	Max Complexity	Ma
Checkpoint1	4 Oct 2021	1	25	11	0.0	3	12.0	3	1.67	0.60	1	

All Methods in Java Project 'test2', Checkpoint 'Checkpoint1'

Class	Method ...	Complexity	Statements	Maximum Depth	Calls
Hello_World	function1()	1	0	0	0
Hello_World	function2()	1	0	0	0
Hello_World	main()	1	1	2	1
Hello_World	main()	1	1	2	1
Hello_World	main()	1	1	2	1

After adding files in SourceMonitor and calculating the checkpoint, display also the “Metrics Summary” like this (below):


SourceMonitor
File Edit View Checkpoint Window Help
Java Checkpoints In Project 'test2*' [Base Directory: 'D:\Tahir\source monitor\']
Metrics Summary For Checkpoint 'Checkpoint1'
Kiviat Graph:
% Comments
Methods/Class
Avg Complexity
Avg Stmts/Method
Avg Depth
Max Depth
Max Complexity
Block Histogram (statements vs. depth):
Help
Copy
Print
Done

Checkpoint Name	Created...	Files	Lines	Statements	% Branches	Calls	% Comments	Classes
Checkpoint1	4 Oct 2021	1	25	11	0.0	3	12.0	3

Parameter	Value
Project Directory	D:\Tahir\source monitor\
Project Name	test2
Checkpoint Name	Checkpoint1
Created On	4 Oct 2021, 09:58:08
Files	1
Lines	25
Statements	11
Percent Branch Statements	0.0
Method Call Statements	3
Percent Lines with Comments	12.0

Optional material:

For these labs students can also use other tools (Google: tools for lines of code). You will get many tools. One of them is SLOCCount.

> **SLOCCount** (pronounced "sloc-count") is a suite of programs for counting physical source lines of code (SLOC) in potentially large software systems. Thus, SLOCCount is a "software metrics tool" or "software measurement tool". SLOCCount was developed by David A. Wheeler, originally to count SLOC in a GNU/Linux distribution, but it can be used for counting the SLOC of arbitrary software systems. SLOCCount can count physical SLOC for a wide number of languages. SLOCCount has some report-generating tools to collect the data generated, and then present it in several different formats and sorted different ways. The report-generating tool can also generate simple tab-separated files so data can be passed on to other analysis tools (such as spreadsheets and database systems).
>
> To use SLOCCount on Windows, you need to first install Cygwin.
>
> **Install Cygwin** **https://cygwin.com/install.html**
>
> To run on Windows, you have to install Cygwin first to create a Unix-like environment for SLOCCount.
>
> **Download and install Sloccount from here:**
> **https://dwheeler.com/sloccount/sloccount-2.26.tar.gz**
>
> Then copy it in: C:\cygwin64\home\"user name" (replace "user name" with the name of your Windows account)
>
> **For further information: read the sloccount user guide**
>
> **https://dwheeler.com/sloccount/sloccount.html**