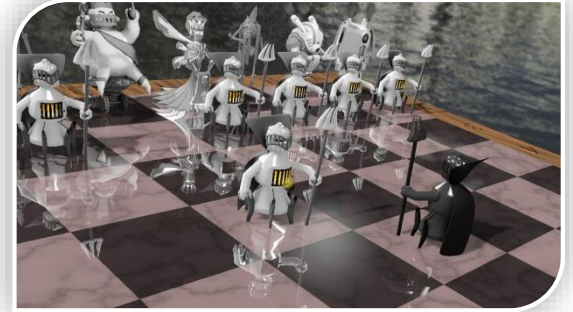
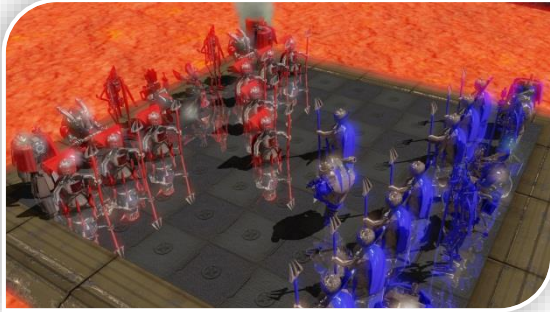


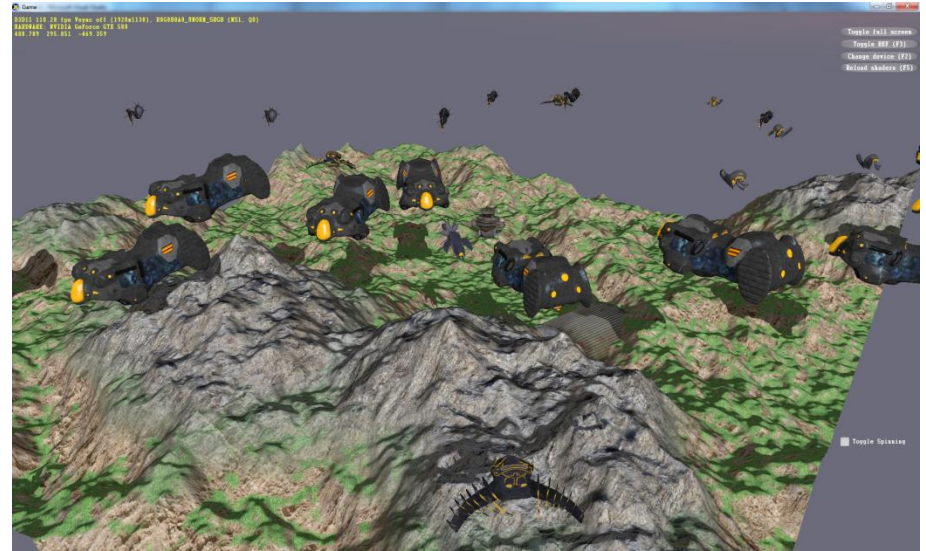
GED Practical Course



tum.3D
computer graphics & visualization










Incoming: Previous Results



- Build a game „from scratch“
 - No existing engine
- Of course we can not write a game engine in one semester...
 - Focus on rendering (DirectX)
 - Also a bit of particle systems, physics, game object management, configurations etc
- But let's start slowly

Index of / /

Name	Size	Date Modified
[parent directory]		
GET_  _FINAL_ROUND_FINAL_FINAL.txt	133 B	1/17/10 7:37:00 PM
PUBLIC_DEMO/		1/25/10 3:32:00 AM
  _PC.zip	2.4 MB	11/9/09 1:29:00 AM
Readme.txt	2258 B	1/17/10 2:35:00 AM
  _FINAL_ROUND_FINAL.zip	99.7 MB	1/17/10 6:11:00 AM
  _FINAL_ROUND_FINAL_FINAL.zip	99.7 MB	1/17/10 7:36:00 PM
  _FINAL_ROUND_FINAL_FINAL_FINAL.zip	99.7 MB	1/17/10 9:48:00 PM
  _FINAL_ROUND_FINAL_FINAL_FINAL_FINAL.zip	99.7 MB	1/19/10 4:34:00 AM
  _FINAL_ROUND.zip	99.7 MB	1/17/10 2:56:00 AM
Sigh - just get the one with the latest date and time stamp and most FINAL_suffixes.txt	133 B	1/17/10 9:49:00 PM

- Software Configuration Management
- Tools to solve problems like:
 - Multiple developers working on the same project
 - Tracking changes
- **Each** project with more than a single developer needs an SCM system
- Benefits for a single developer, too: Rollback, Branches, etc.

- Centralized: one server, multiple clients
- Each change must be „committed“ to the server
 - Each „Commit“ increases the revision number
- All changes are „atomic“: repository always consistent
- Each revision can be restored
 - Tracking changes
 - Reverting errors

- Sourcecode history is stored on the server
- You work on a „working copy“
 - Basically a copy of a revision
 - May contain local changes
 - „Per file“ basis
 - So only a part of your working copy might be up-to-date if you choose so

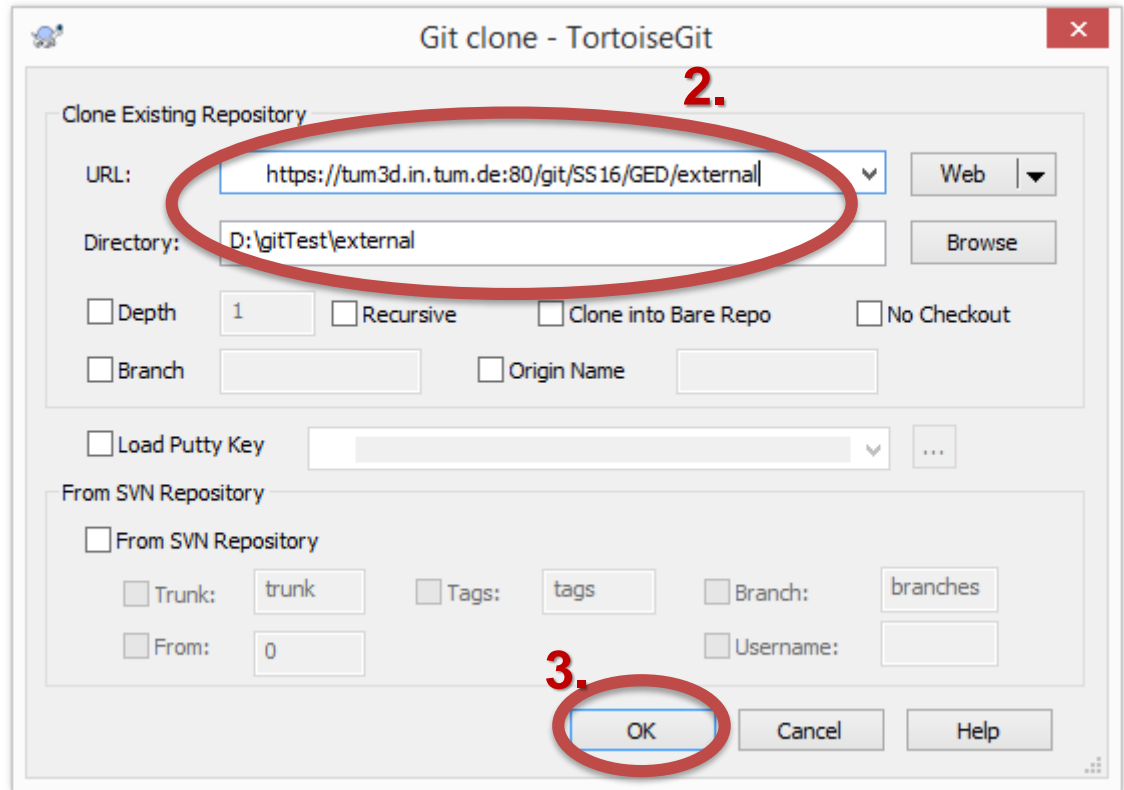
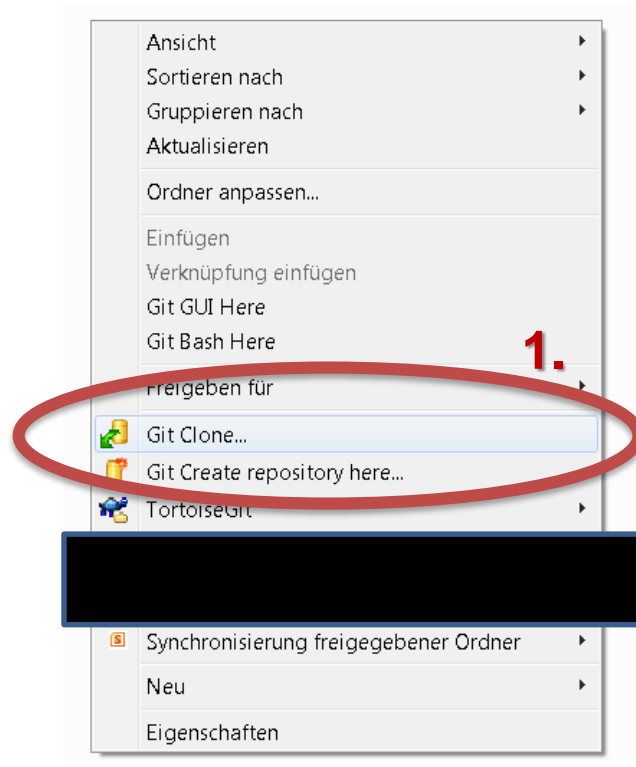
- *Every* file of the project needs to be in your Git repository
 - Everything needed to build the program
 - You will work on multiple PCs!
 - Check regularly if this is the case
- Do *not* add files created from your code to your repository
 - *.exe, *.dll
 - Automatically created and temporary files
 - *.opensln, *.sdf, etc...
 - See the first assignment for details

- „Commit early, commit often“
 - So you can also trace small changes with great impact...
- update before you start working
- update before and after a commit
 - Reduces conflicts
- Ignore list: Add Debug, Release, .ncb, .suo, etc.
- **Always** include a **meaningful** Commit-Message
 - Helps you track down changes

- 2 Repositories:
 - <https://tum3d.in.tum.de/git/SS17/GED/external>
 - Contains Meshes, Textures, Examples, Libraries, Assignments, Slides, etc
 - Regular updates!
 - <https://tum3d.in.tum.de/git/SS17/GED/<username>>
 - Your source code
 - Create at <https://tum3d.in.tum.de/ged/register.php>
- Help!
 - Git: <https://git-scm.com/doc>
 - TortoiseGit: <https://tortoisegit.org/>

- You can checkout your repository, to a folder of your choice, say `.\ged\
 - Do not forget to commit and push before logging out!`
- Checkout `external` to `.\ged\external`
 - Update before you start working!

- Checkout with TortoiseGit

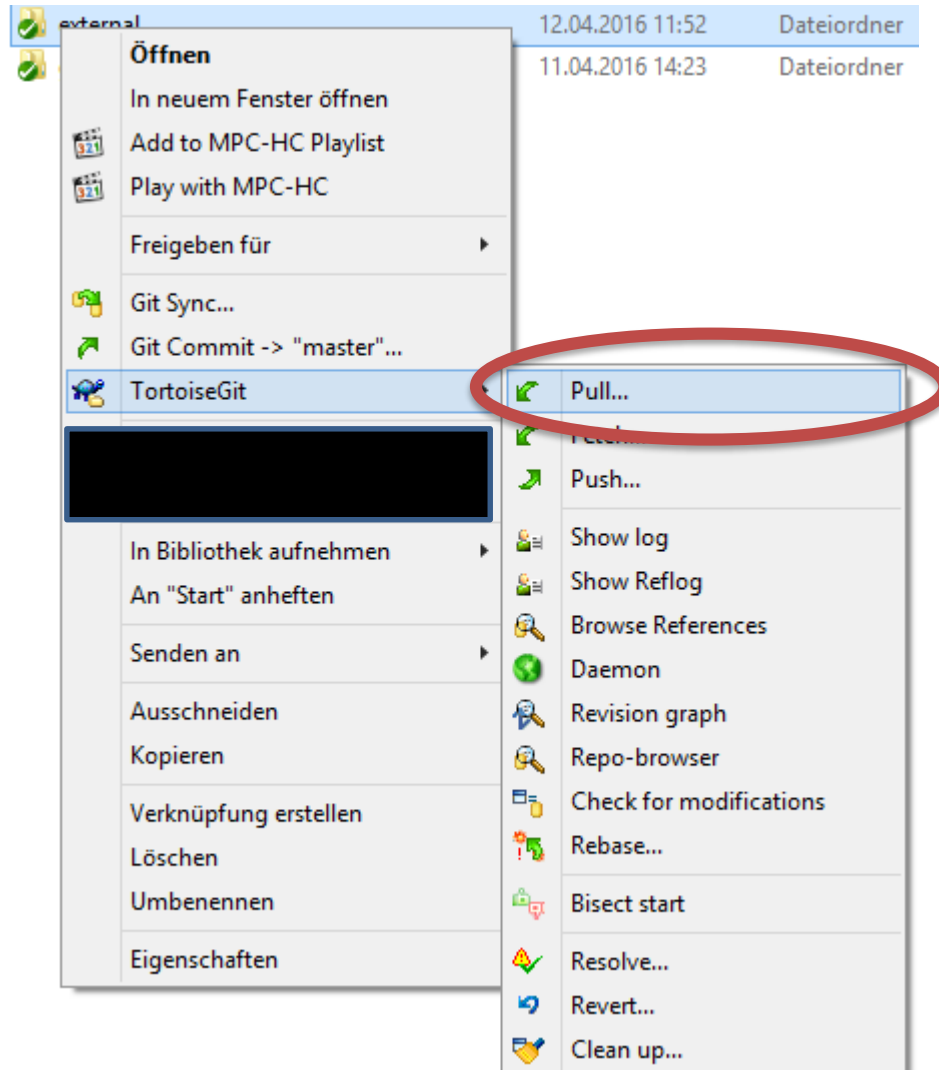


- Checkout external and <username> to the same folder!

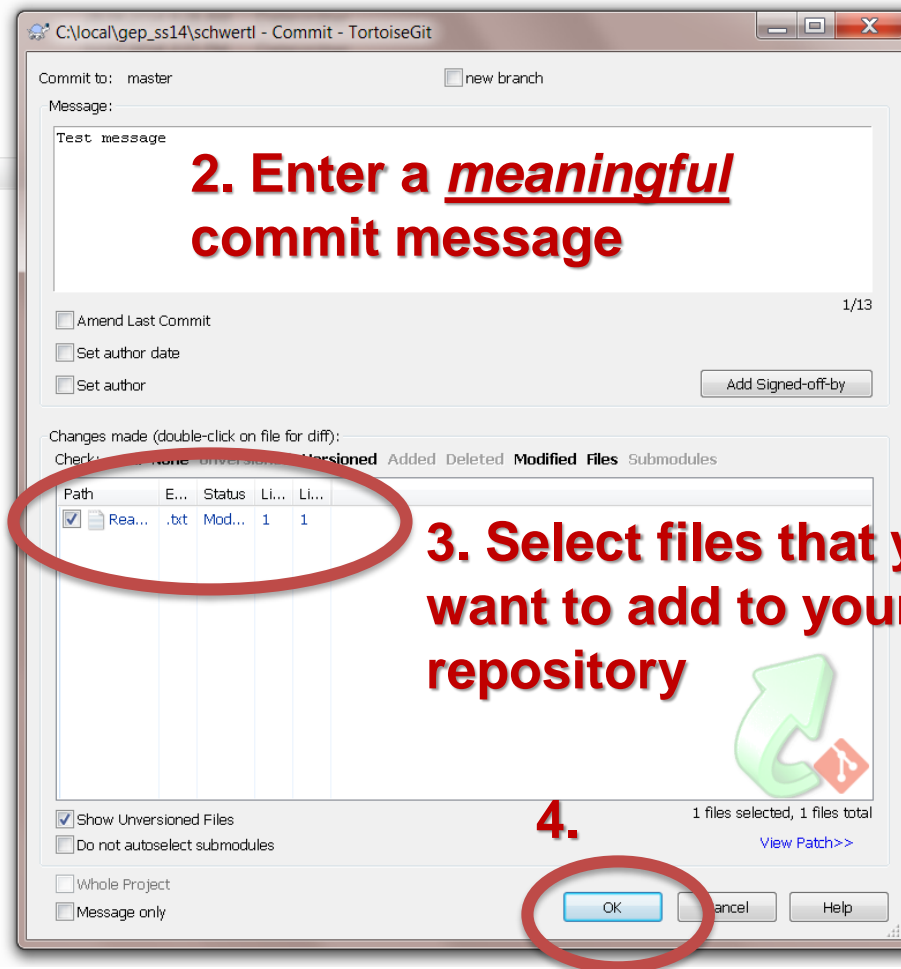
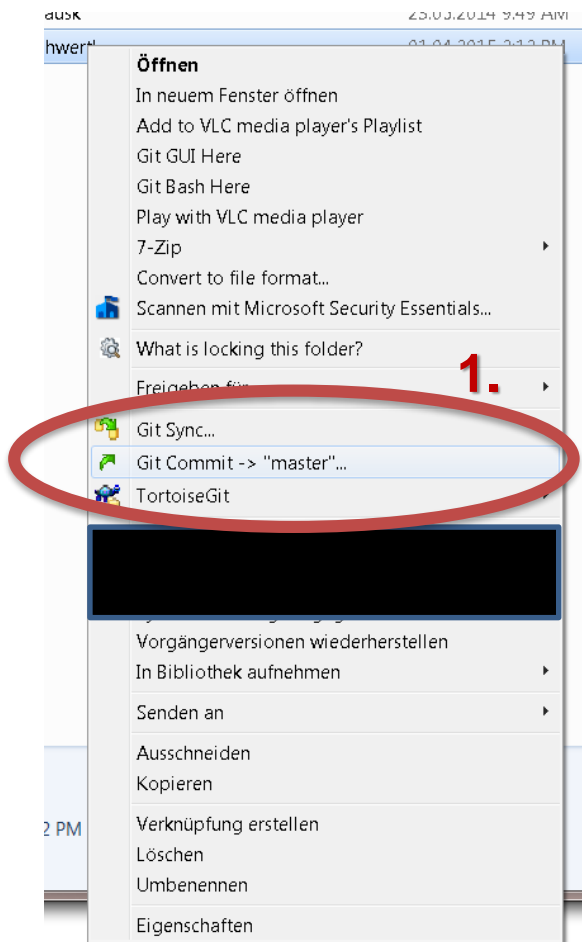


- Only work in your folder, don't change anything in external

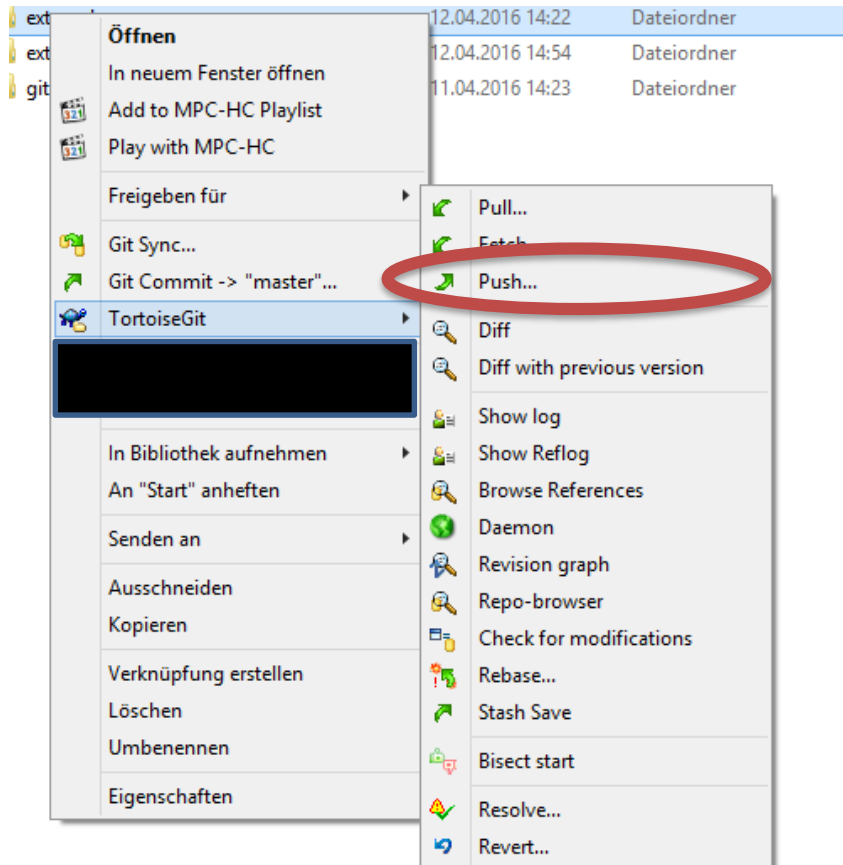
- Update local repository with TortoiseGit



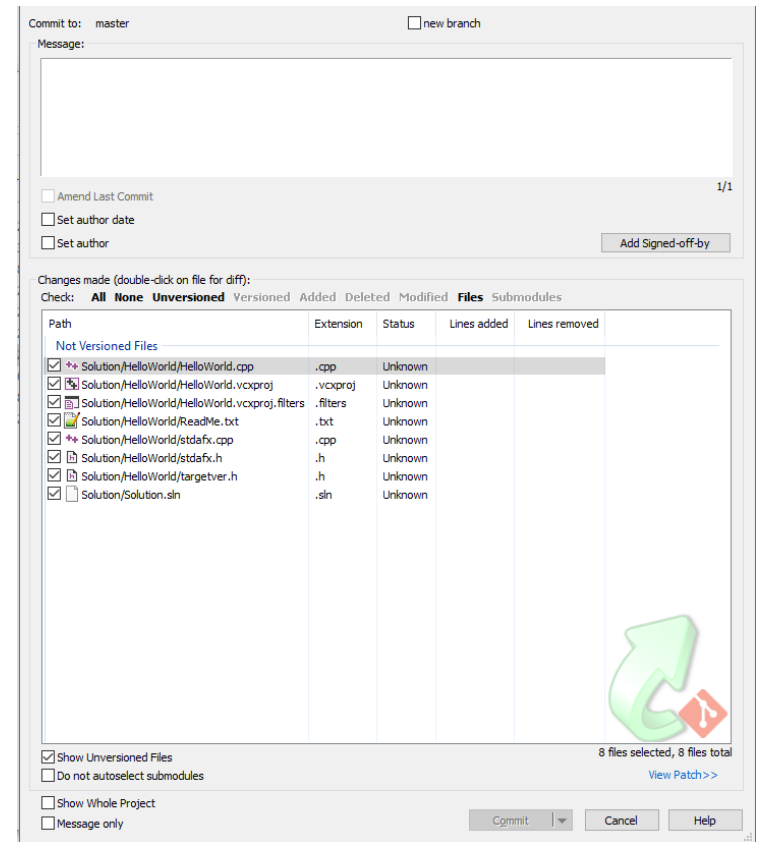
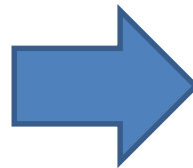
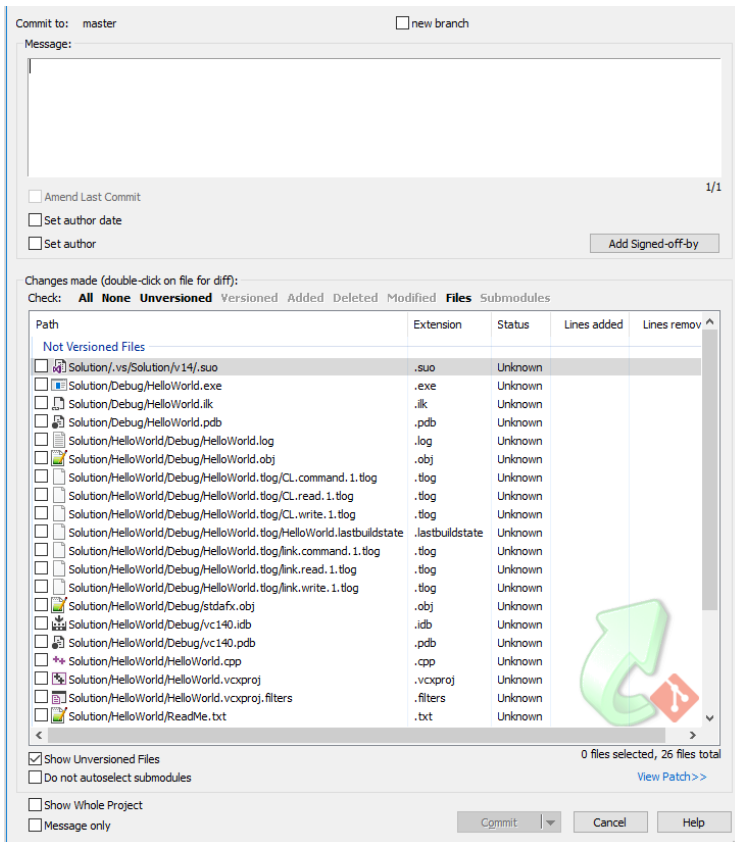
- Commit
- Adding changes to the local repository



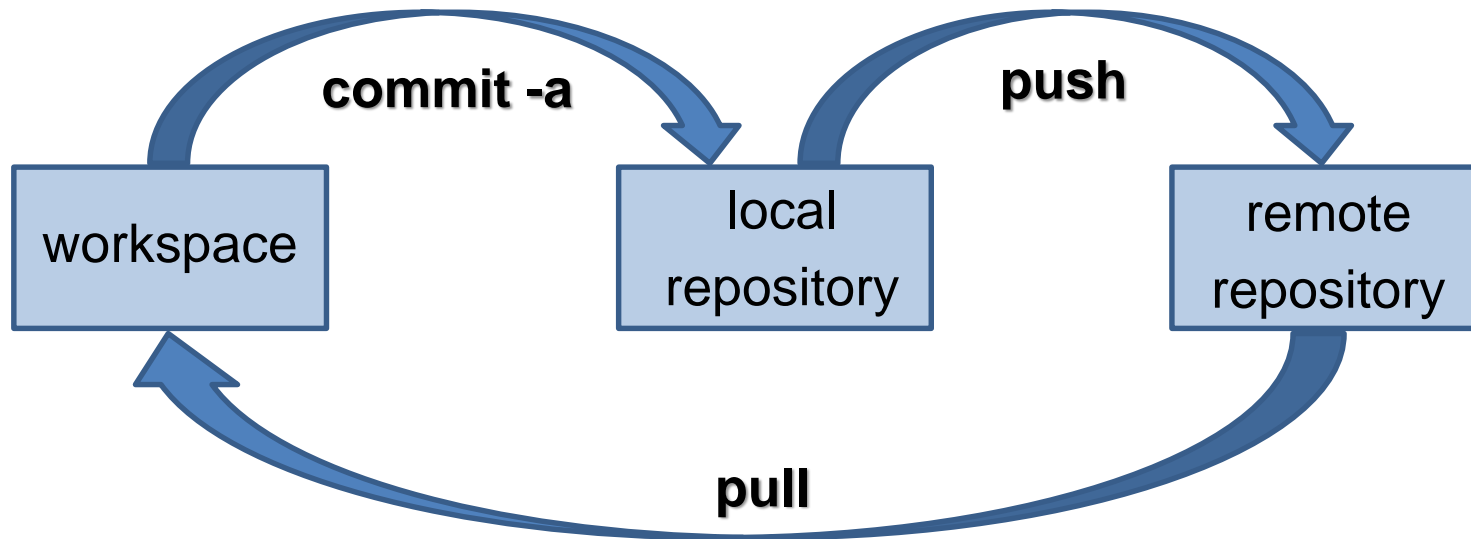
- Push
- To upload your (local) commits to the server



- “.gitignore” textfile specifies intentionally untracked files to ignore



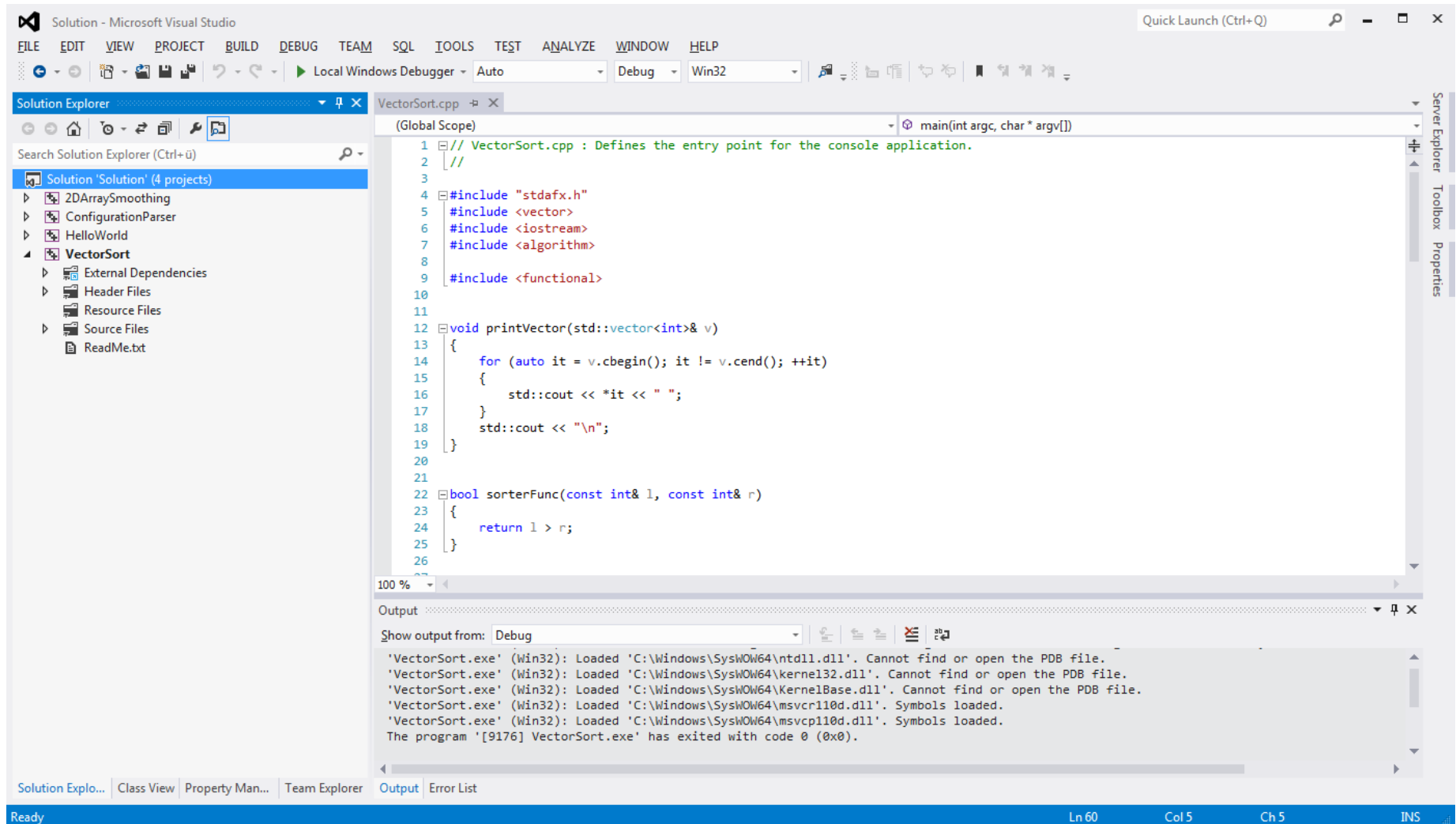
- Simplified overview



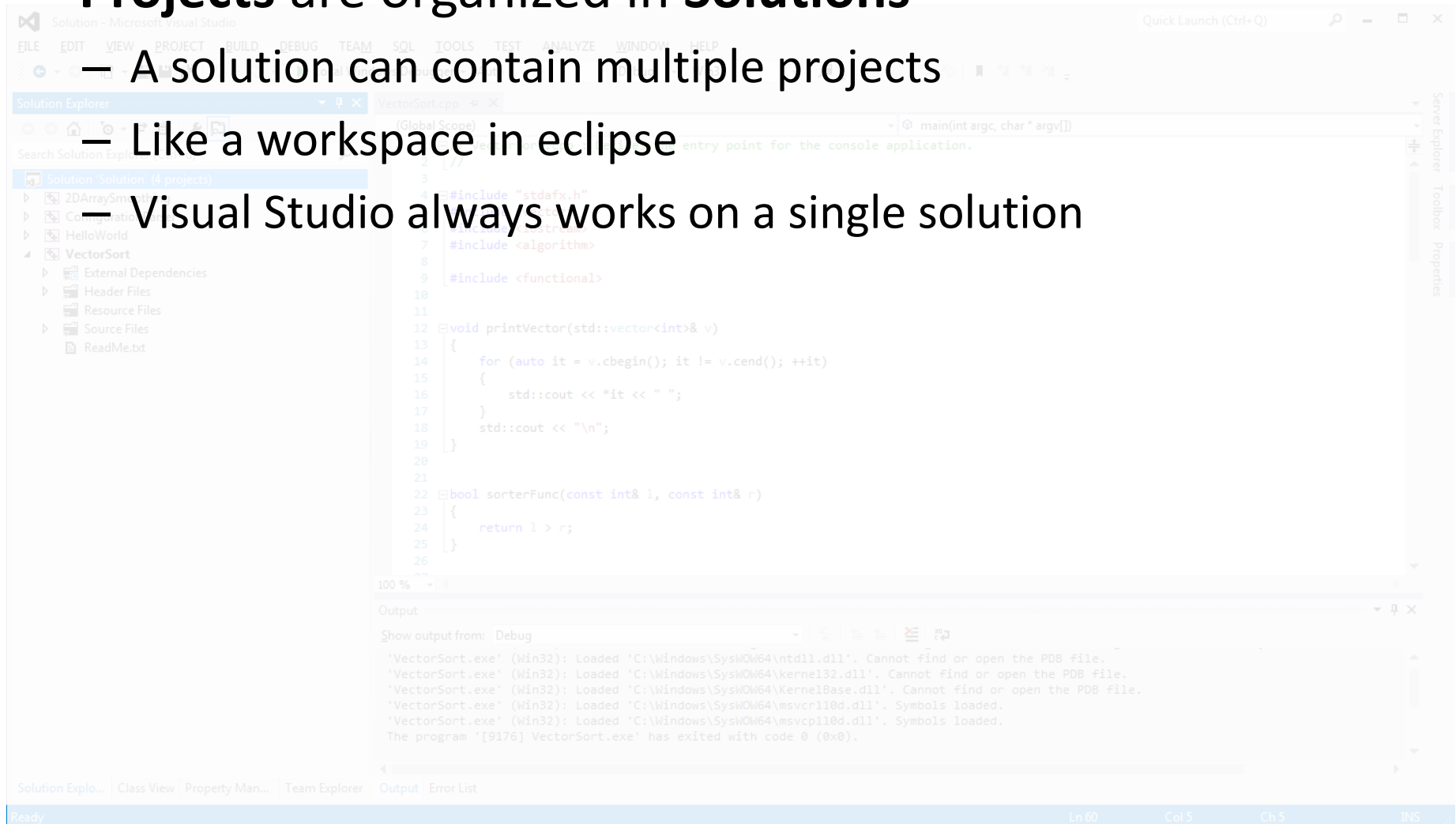
- More information: <https://git-scm.com/book/en/v2>

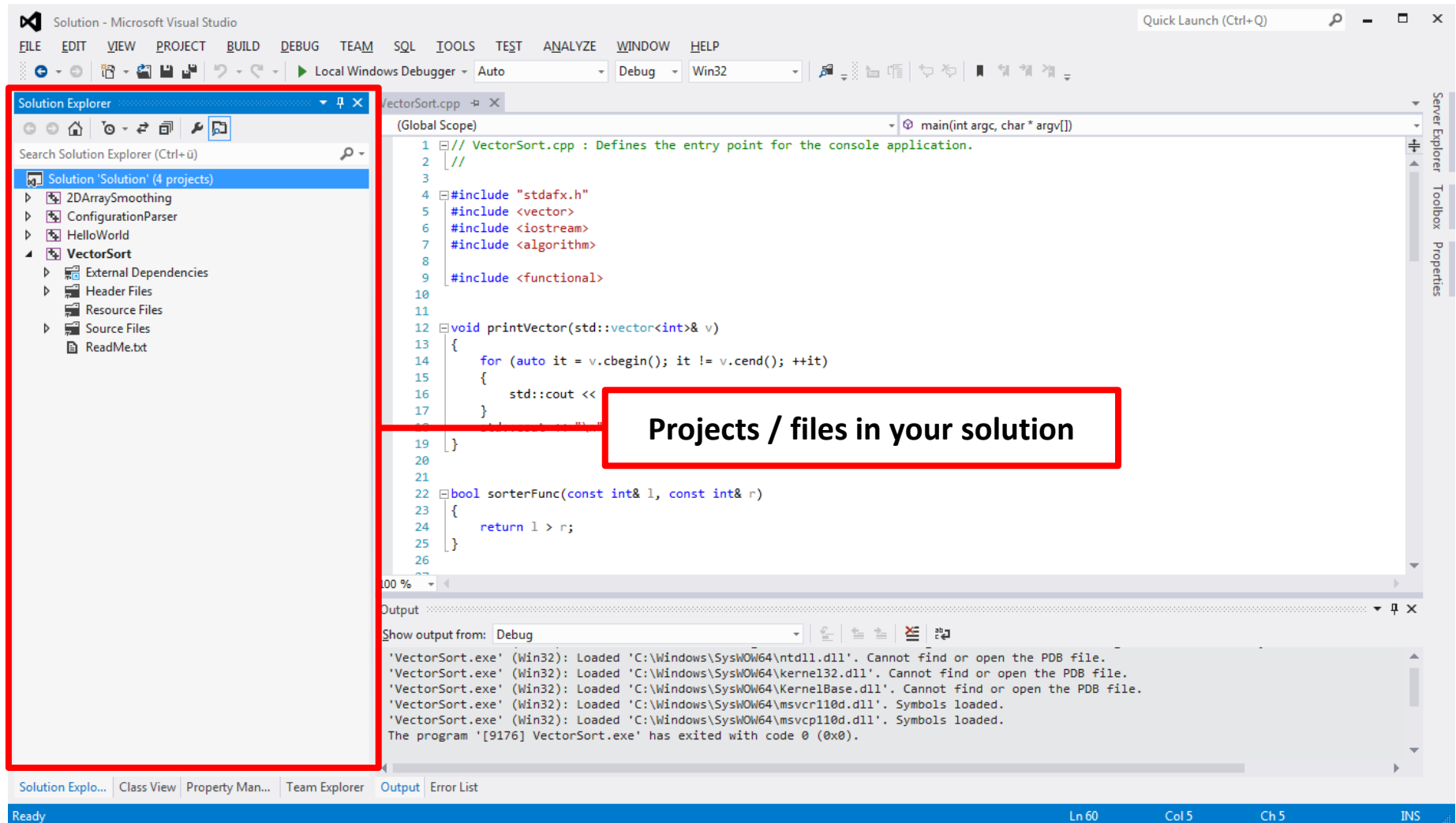
- Microsofts IDE for C++, C#, VB, F#...
 - Not just a compiler
 - Editor w/ syntax highlighting, code completion etc
 - Debugger
 - Several other tools you won't need

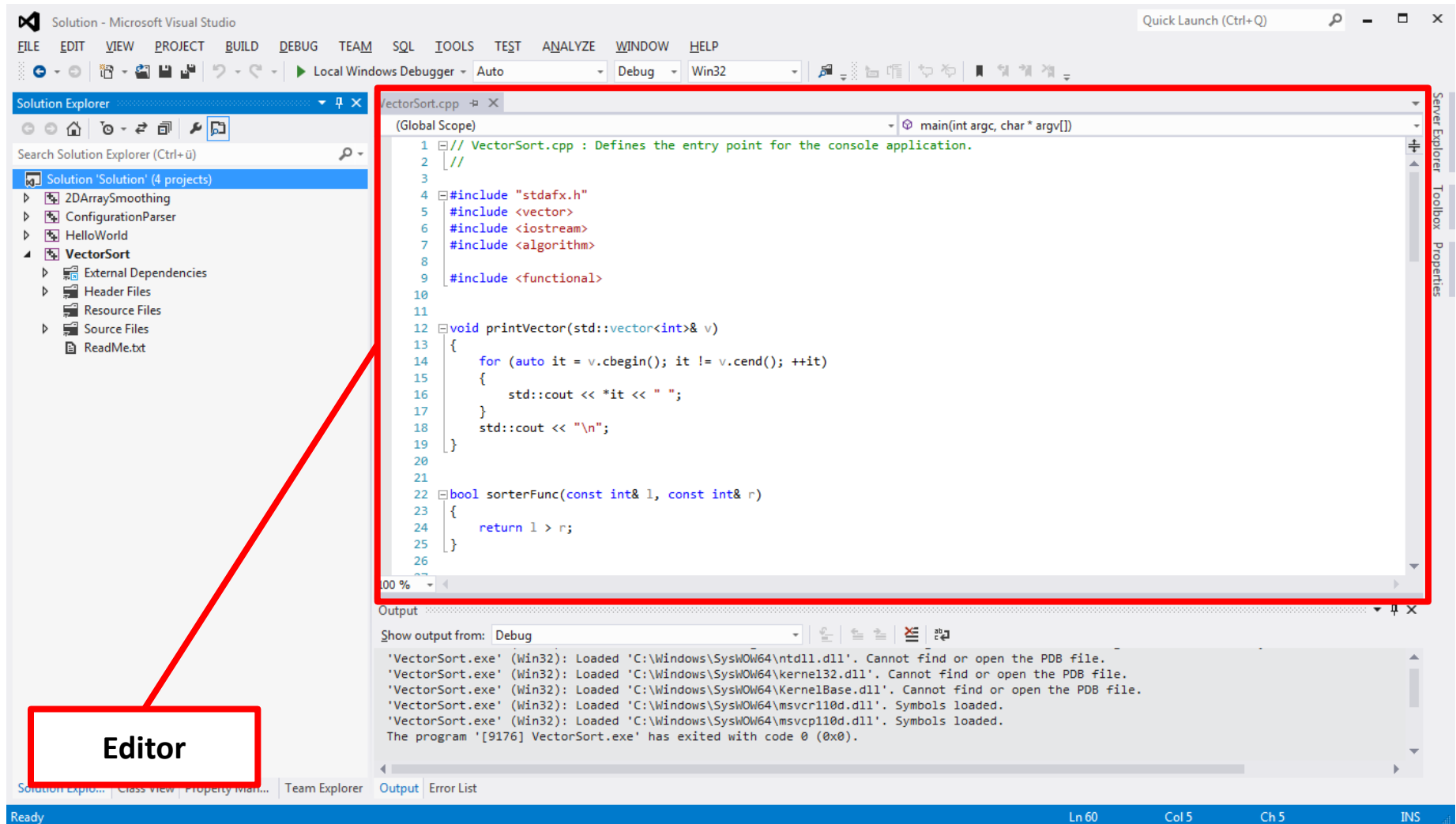




- **Projects** are organized in **Solutions**
 - A solution can contain multiple projects
 - Like a workspace in eclipse
 - Visual Studio always works on a single solution



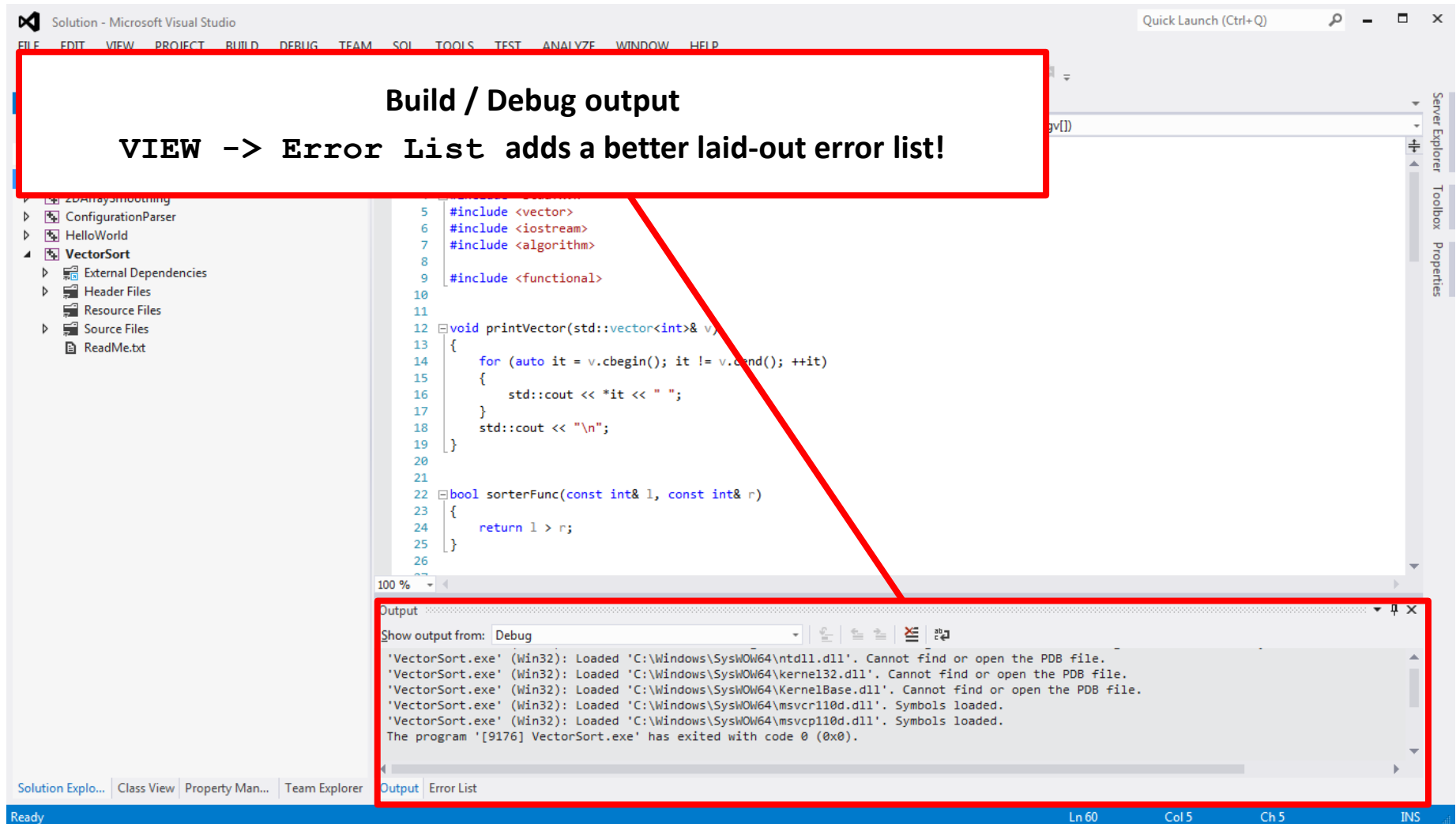


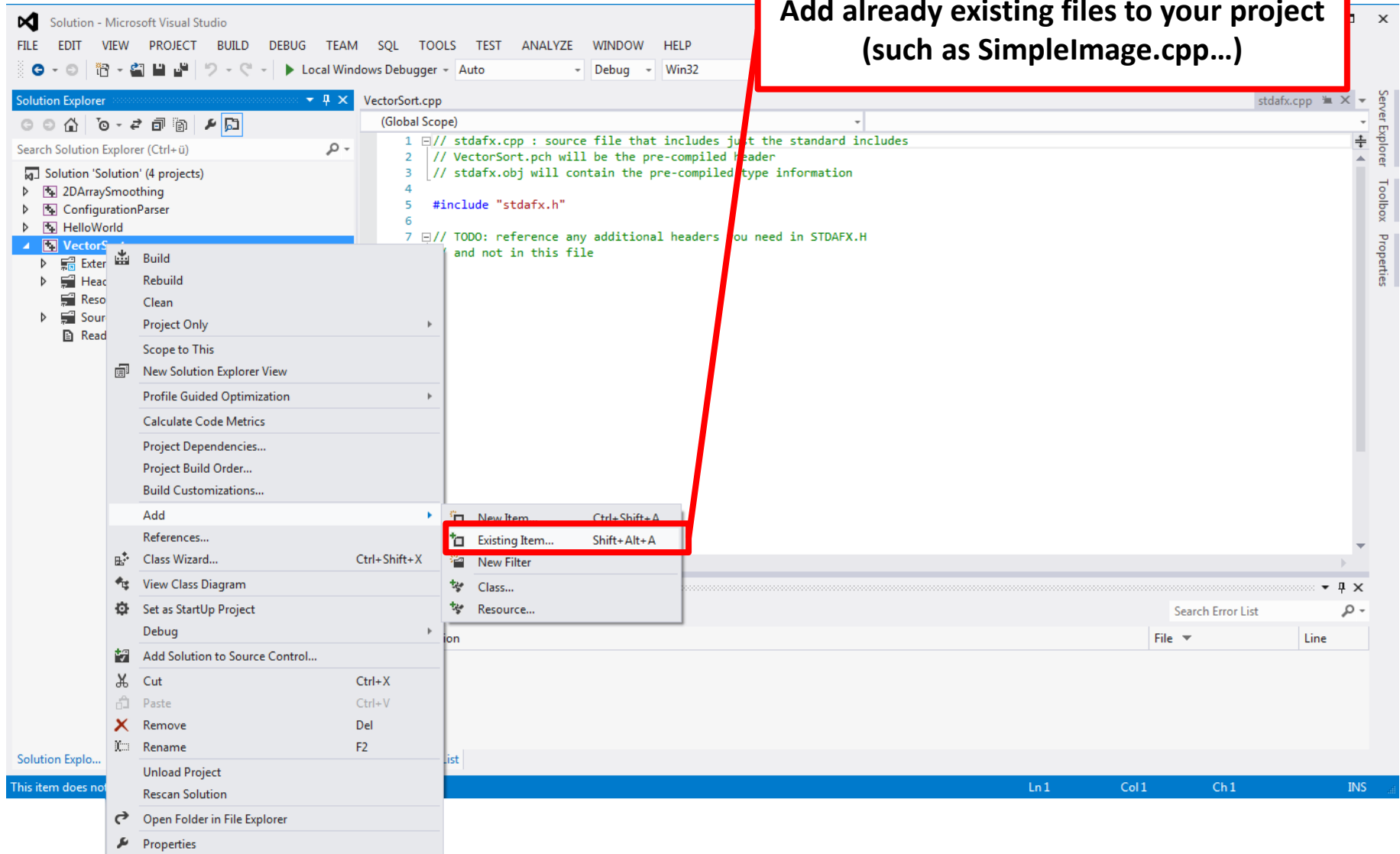


Editor

Build / Debug output

VIEW -> Error List adds a better laid-out error list!

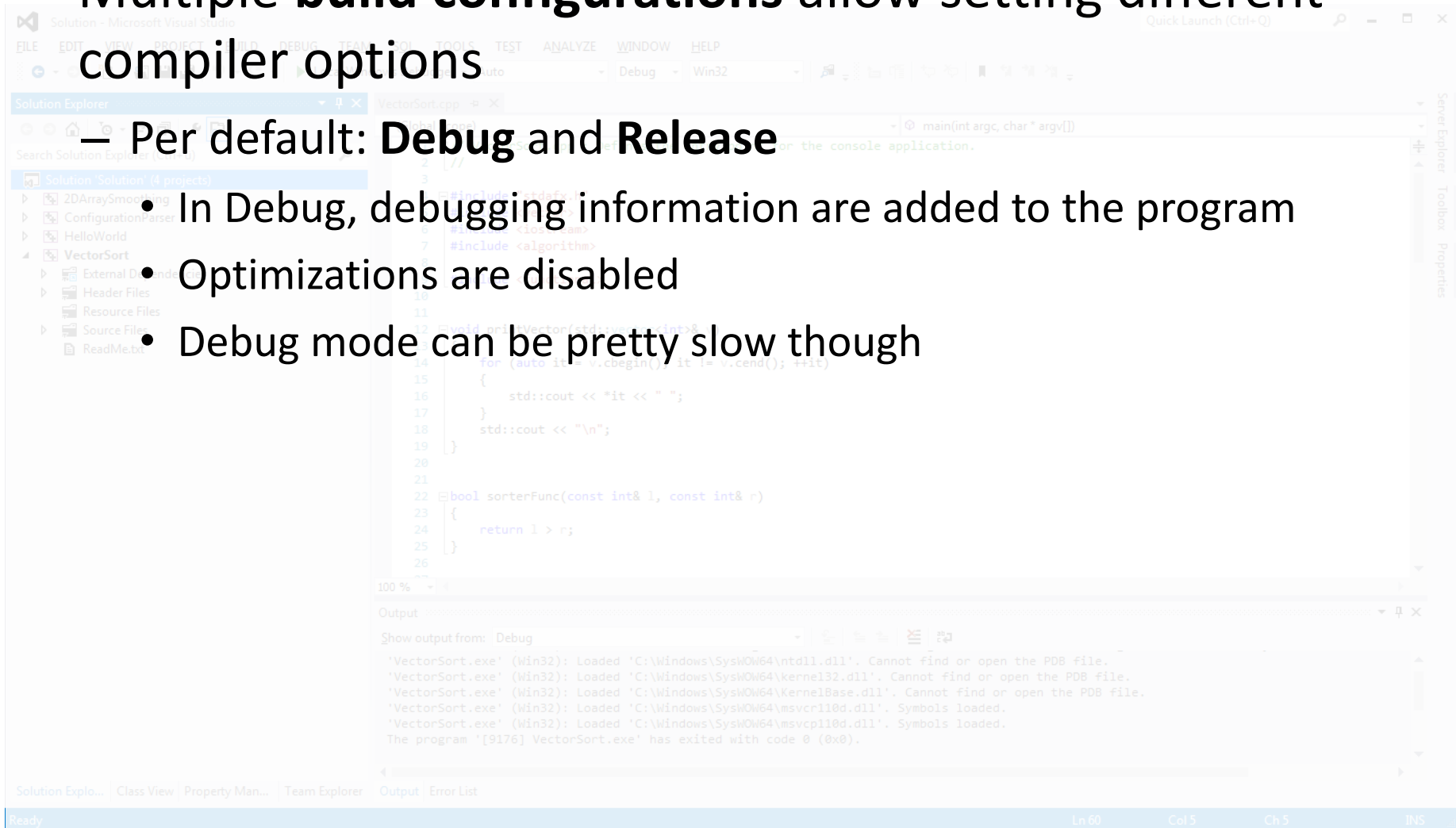


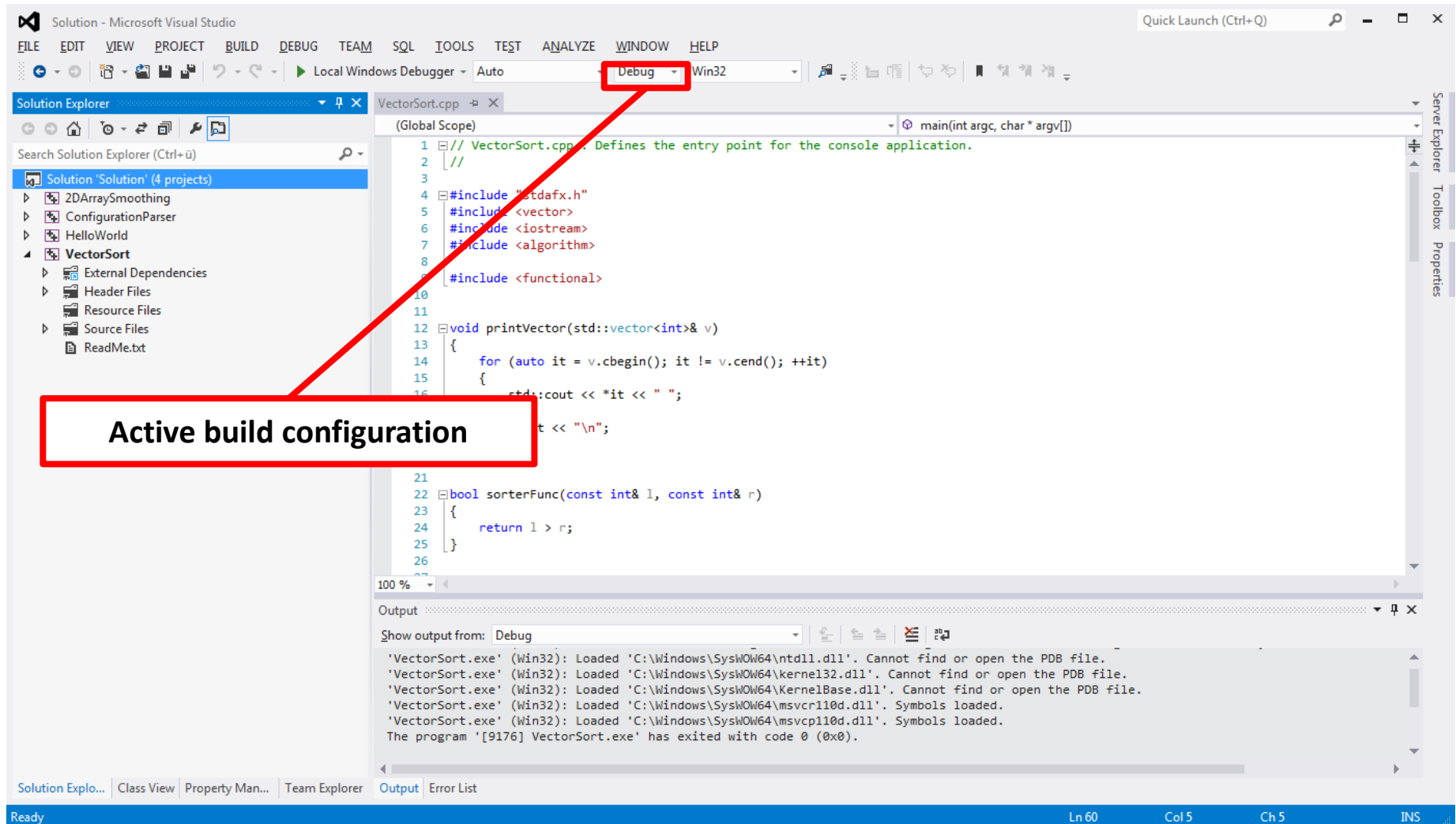


- Multiple **build configurations** allow setting different compiler options

— Per default: **Debug** and **Release**

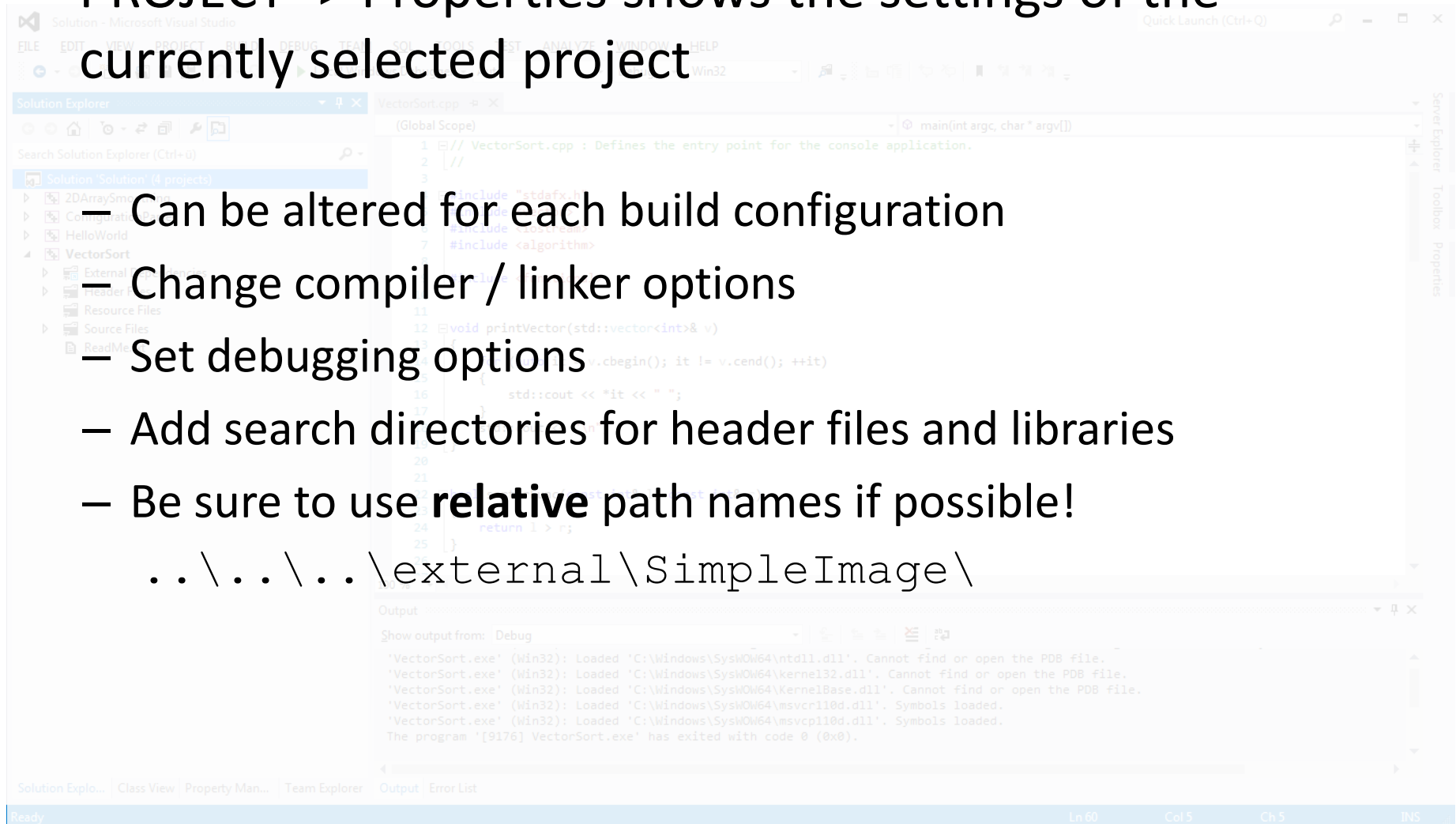
- In Debug, debugging information are added to the program
- Optimizations are disabled
- Debug mode can be pretty slow though

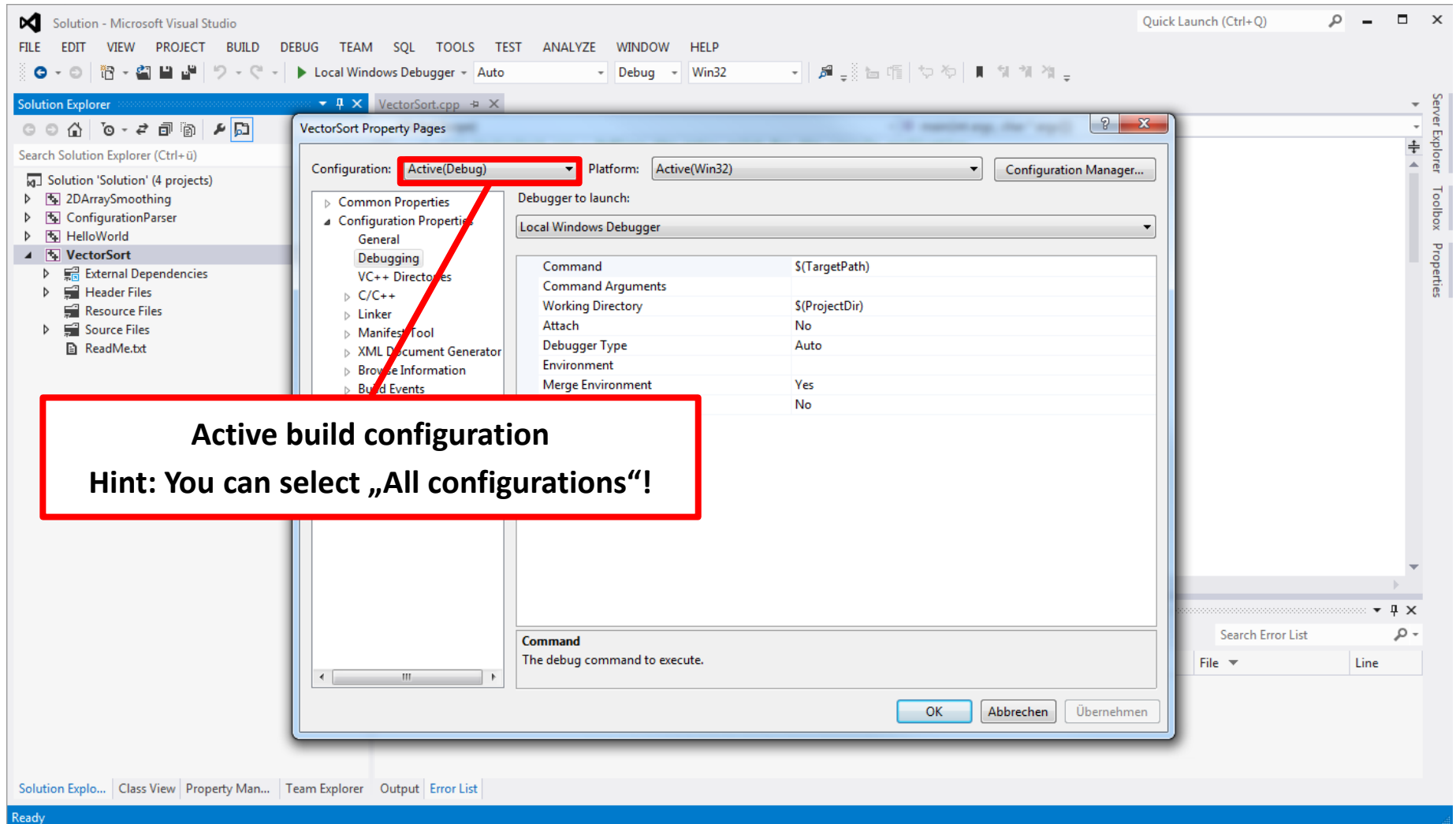




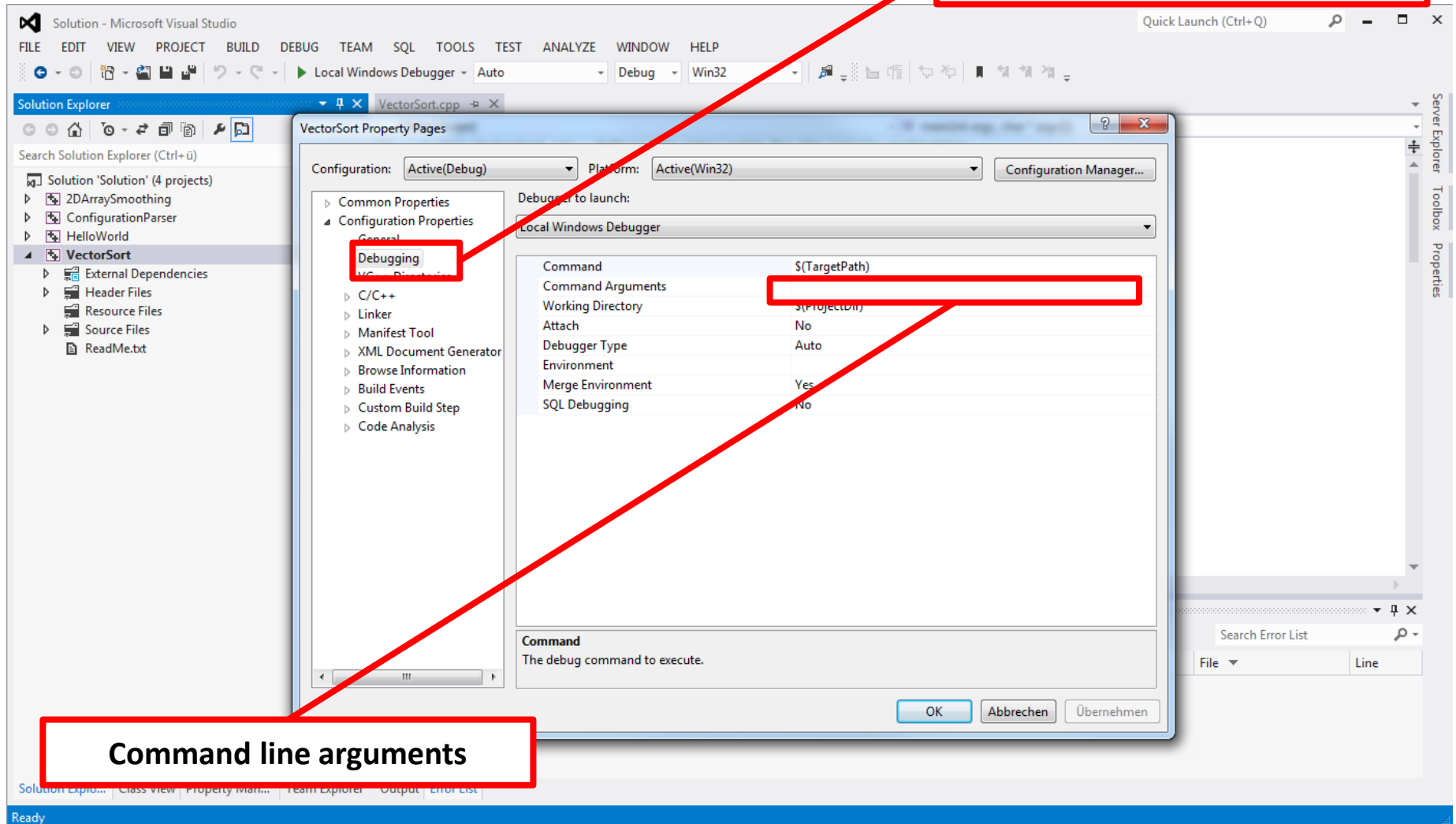
- PROJECT -> Properties shows the settings of the currently selected project

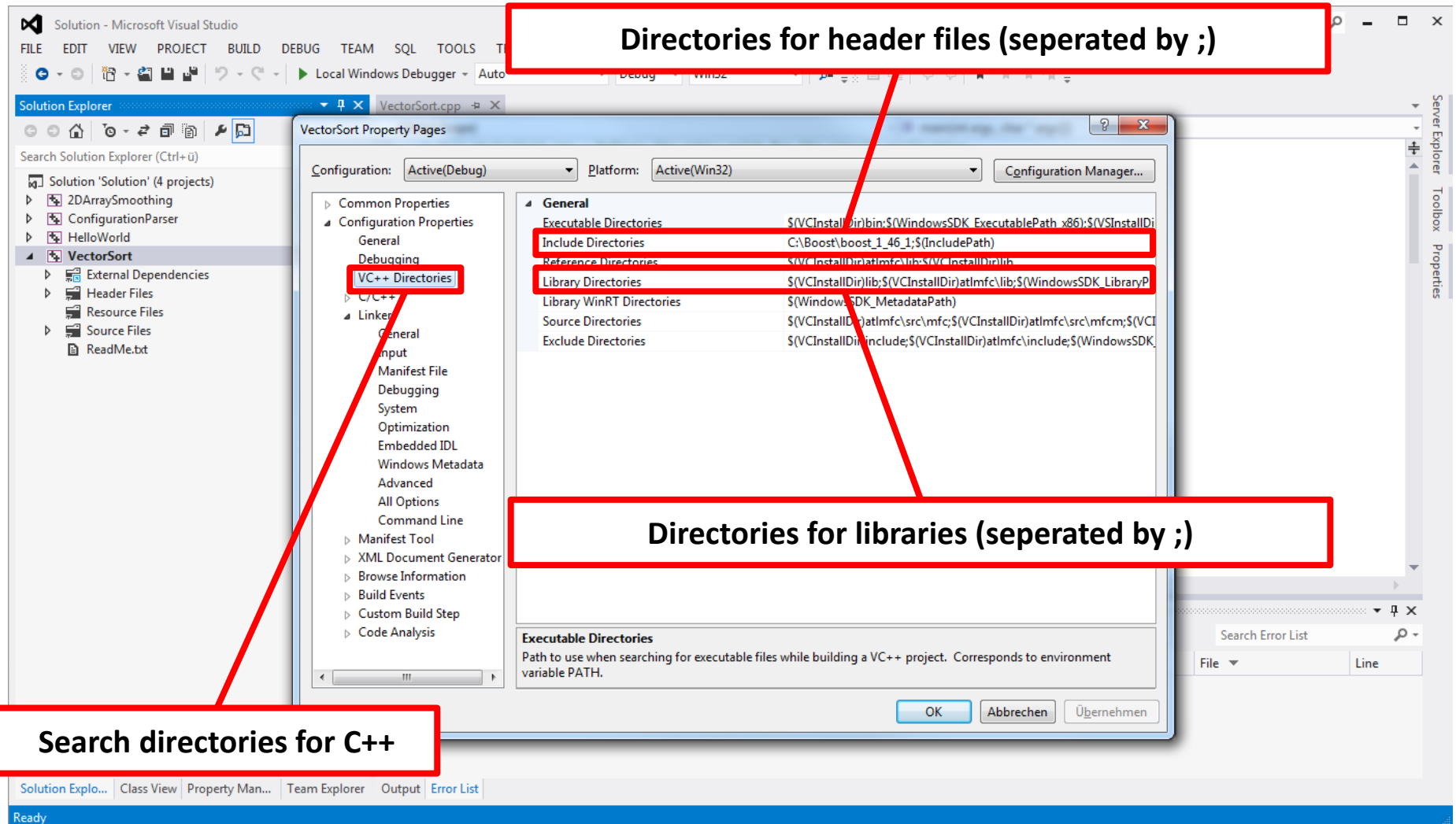
- Can be altered for each build configuration
- Change compiler / linker options
- Set debugging options
- Add search directories for header files and libraries
- Be sure to use **relative** path names if possible!
 `..\..\..\external\SimpleImage\`

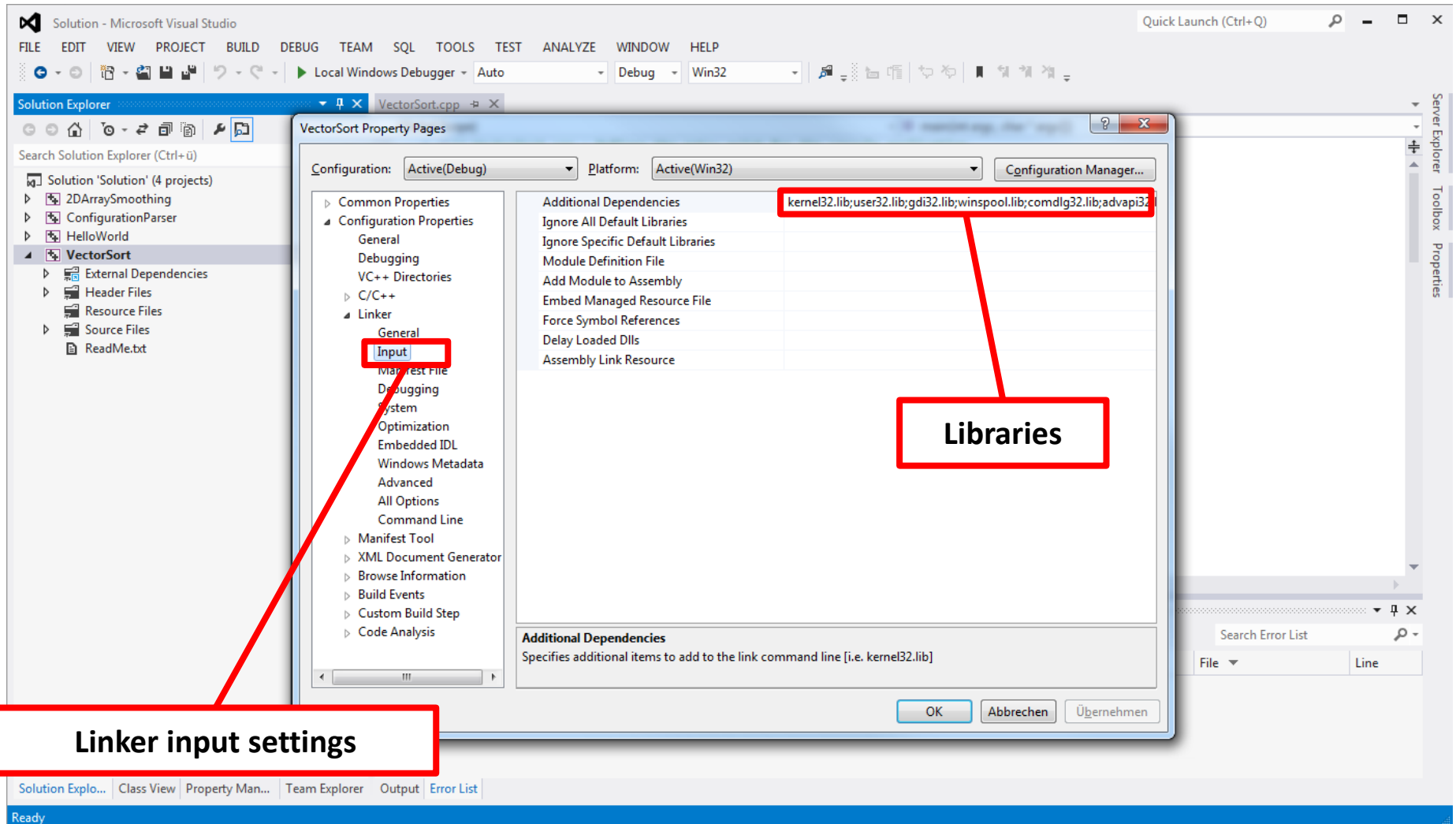




Debugging options







Questions?

You can also use our Q&A forum for further questions.