Unit 0

C++ Workshop

Introduction

Course Outline

Unit 1. Brickles Specification and Analysis Model
- Understanding frameworks
- Interfacing frameworks

Unit 2. Component Libraries and Design Heuristics
- Designing for evolution
- Standard design patterns

Unit 3. Brickles Design - Increment 1
- Design details of a first version of Brickles
- Models
- Sample code
- Programming exercises

Unit 4. Coding Standards and C++ Language Issues
- Coding Standards and conventions

Unit 5. Increment 2 Design and Implementation
- Design details of the second version of Brickles
- Models
- Sample code
- Programming exercises

Unit 6. Enhancements
- Optimizations
- New features

Unit 7. Course Summary

Course Objectives

The students will:

- have hands-on experience writing C++ programs.
- relate design concepts and patterns to their implementation in C++.
- design and implement enhancements to a C++ system.

The student will NOT:

- learn Windows programming.

Activities

- Understand an existing object-oriented design.

- Implement selected portions of that design.

- Design enhancements to the system.

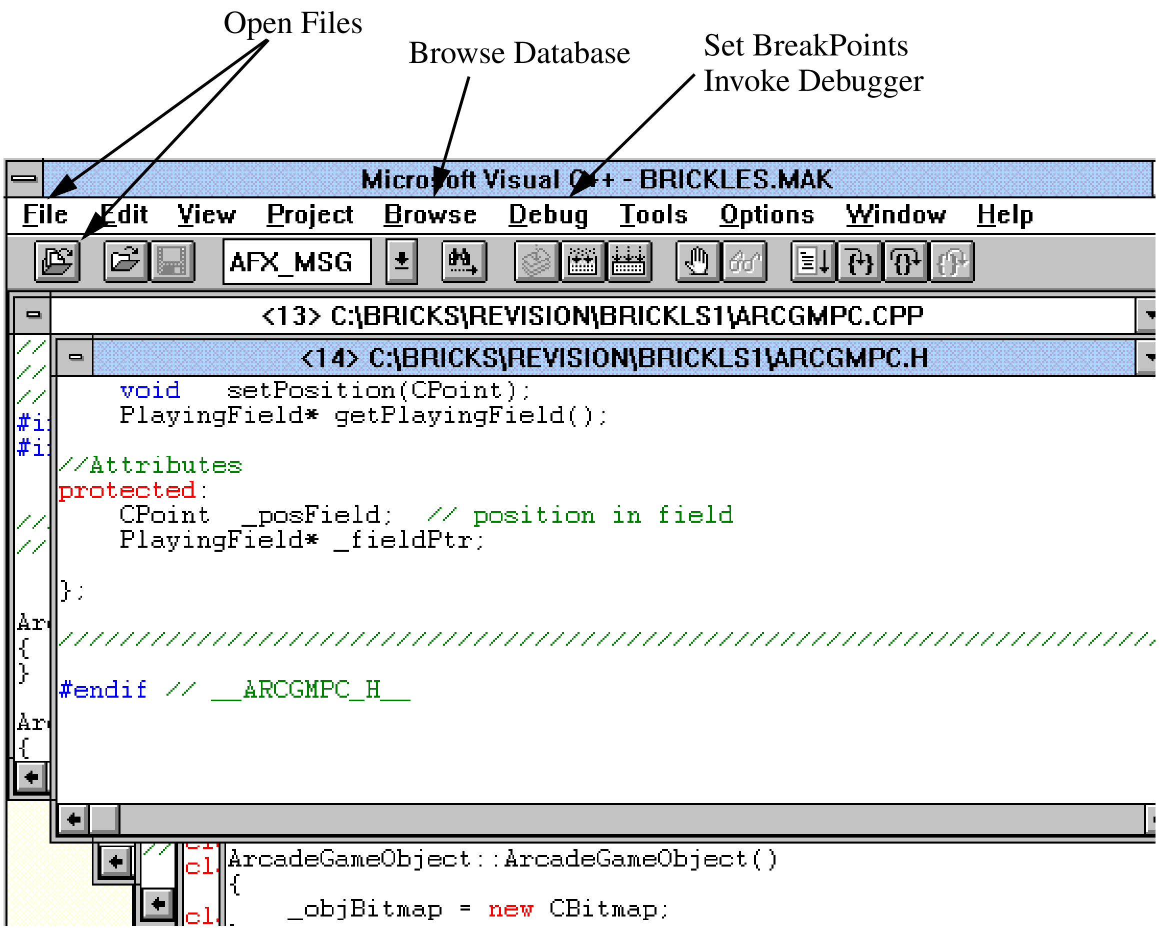
- Implement one of those enhancements.

Note: The course uses the Microsoft Visual C++ environment and the Microsoft Foundation Classes but the majority of the course is directed at “standard” C++. The MFC Library will be used as an example of a framework, we will cover as little of the MFC library as possible.

C++ Concepts

- Constructors/destructors
- Pointers and references
- Inheritance
- Inclusion polymorphism
- Templates
- Exceptions

Visual C++ Development Environment



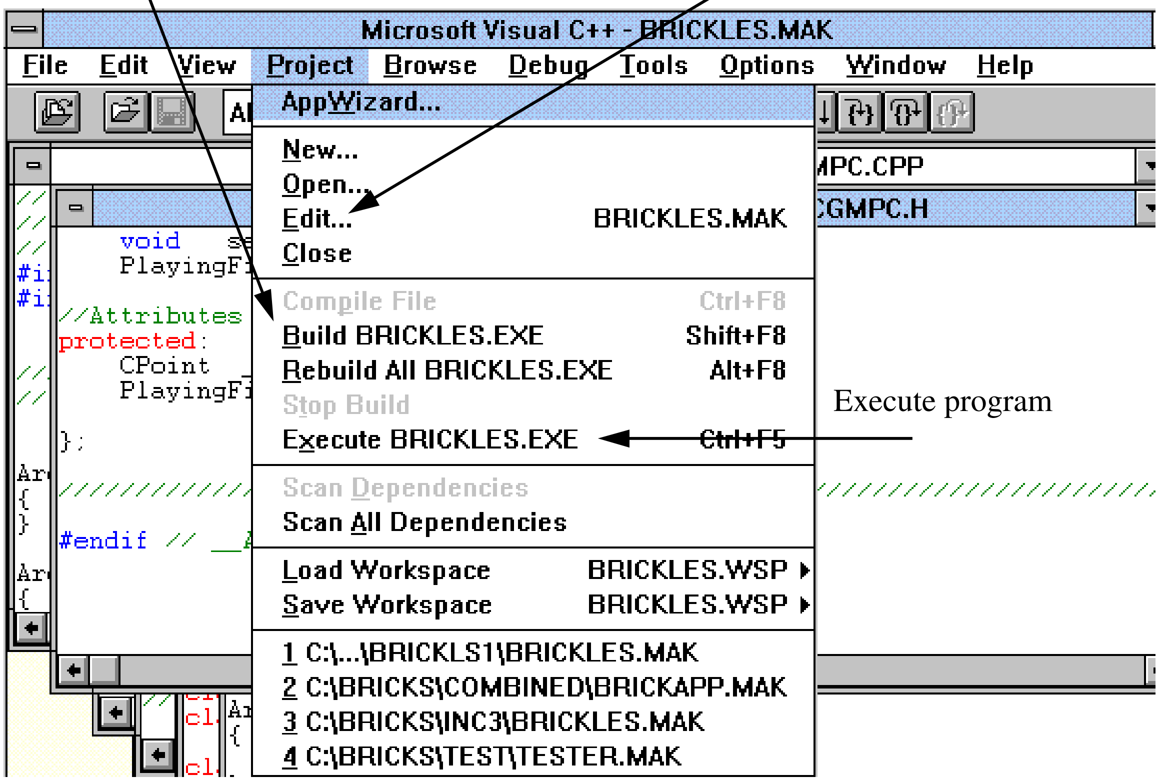
- The development environment provides access to a multi-window editor, Make/Build facility, application browser and the debugger.

- The editor provides highlighting of keywords.

Visual C++ Build Menu

Compile out of date files

Add files to the MakeFile



Execute program

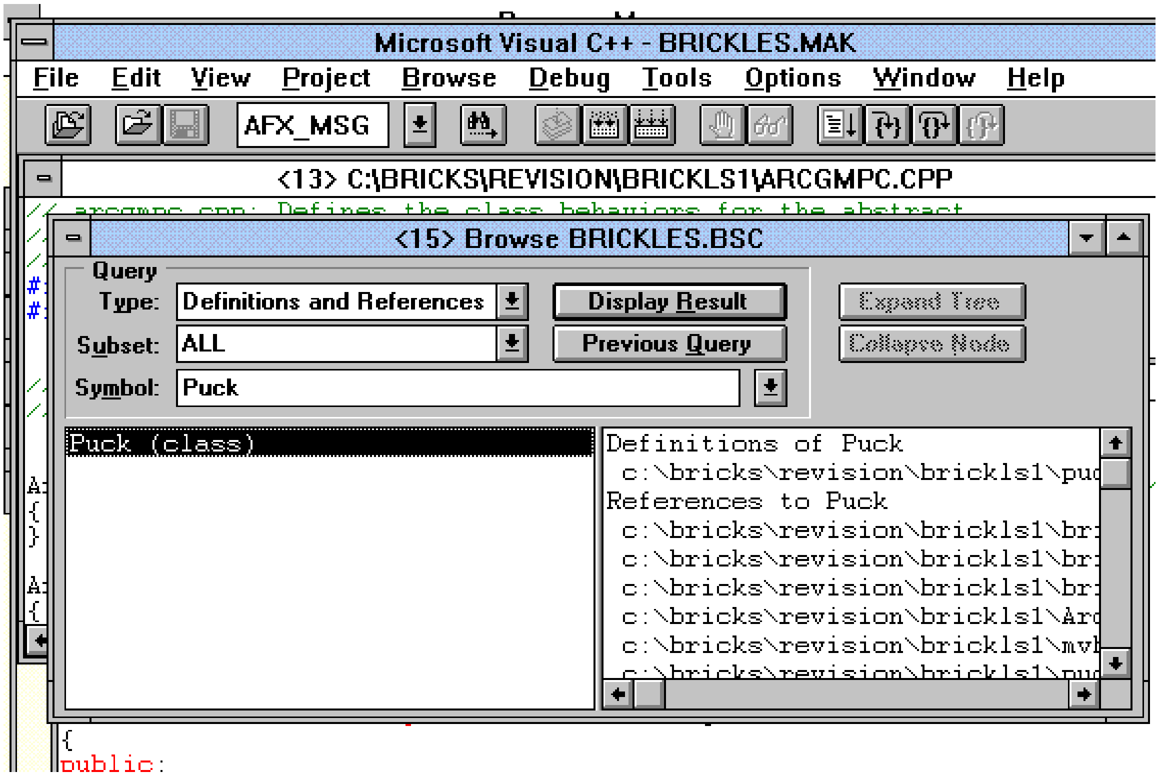
- Edit Brickles.Mak

  Add/Delete files from the Makefile

- Build Brickles.Exe

  Compiles all files that are out of date

Browse Application Database

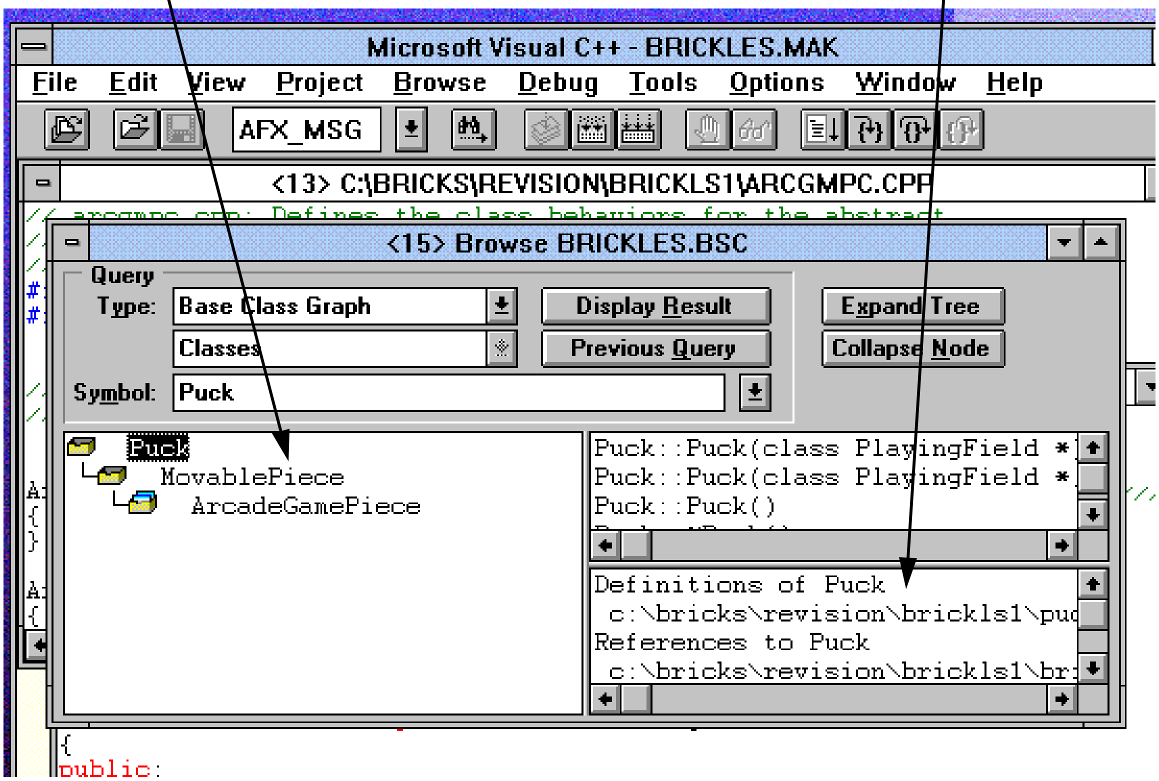


- Set TYPE to the type of display desired
- Set SYMBOL to the name to be viewed
- Select DISPLAY RESULTS
- Windows show
  - Callers and Receivers of messages
  - Class Hierarchies
  - Methods by class

Application Database Browser

Inheritance Tree

Locations of references
and definitions for symbols



- The browser shows the inheritance tree.

- Other windows show:

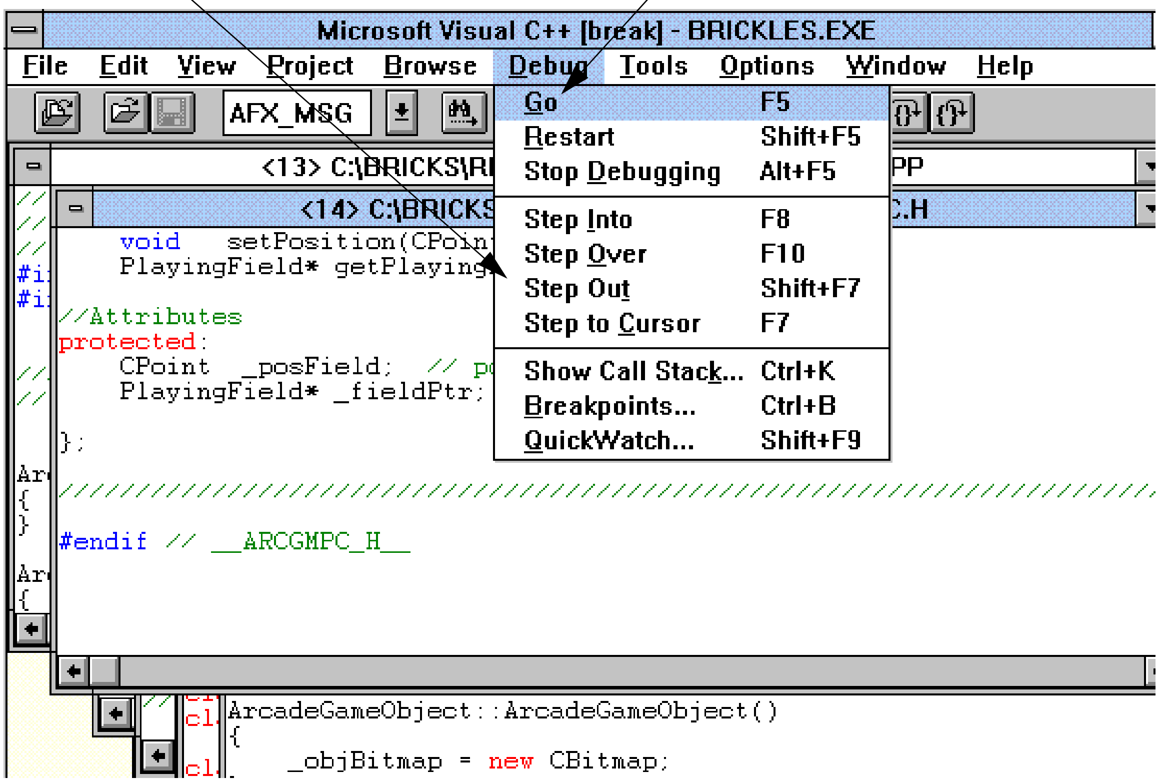
  Definitions and references

  Class members

Visual C++ Debugger

Variety of execution modes.

Execute until a breakpoint is encountered.



- Compile your program.

- Execute by choosing GO from the Debug menu.

- At a breakpoint, use Step ... to examine details of a method, skip over a single message, compute down to the cursor location.

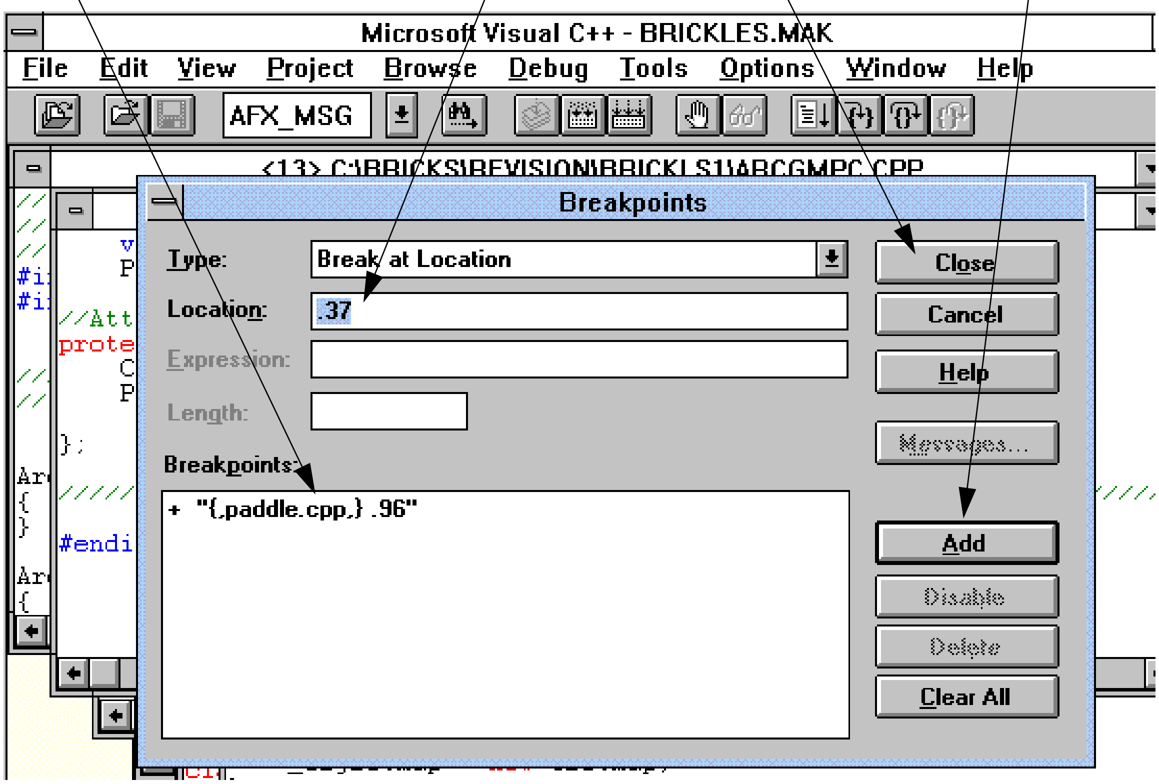
Visual C++ Debugger

Existing breakpoints

Location of cursor in file window

Close saves the current breakpoints

Sets a breakpoint at current cursor location.



- Move edit cursor to the position in a file where the breakpoint is to be set.

- Press **Add** button to add the breakpoint at **Location** to the **Breakpoints** list.

- **Close** will close the window keeping all currently set breakpoints. **Cancel** will close the window but will cancel any breakpoints set since the window was opened.

Exercises

- Execute the Increment 1 version of Brickles. Use the left button of the mouse to display a dialog box from which you may quit the game.
- Use the browser database for Increment 1 to find all references to the Puck class.
- Using the debugger, set breakpoints in the Move method of the Puck class. Add the value of _posField to the Watch list. Execute the program and observe the chanes in values.