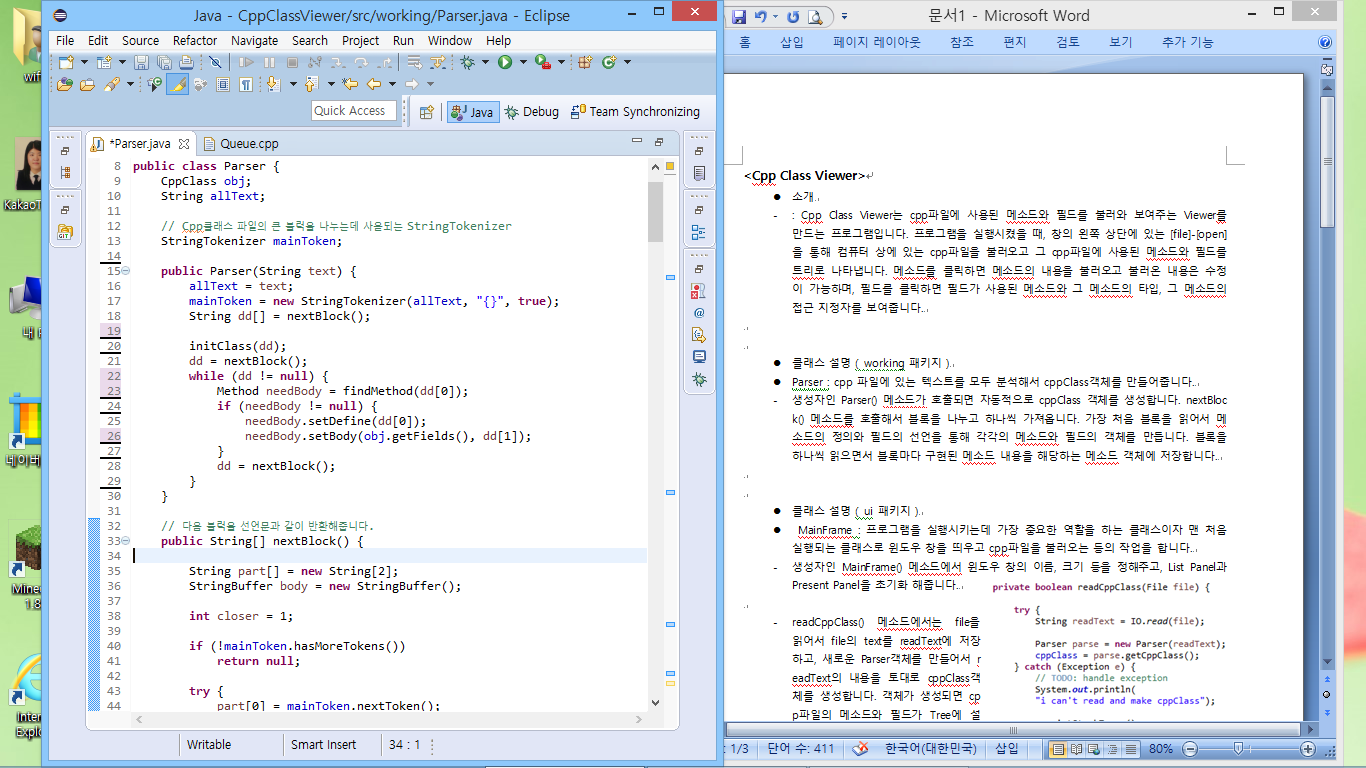
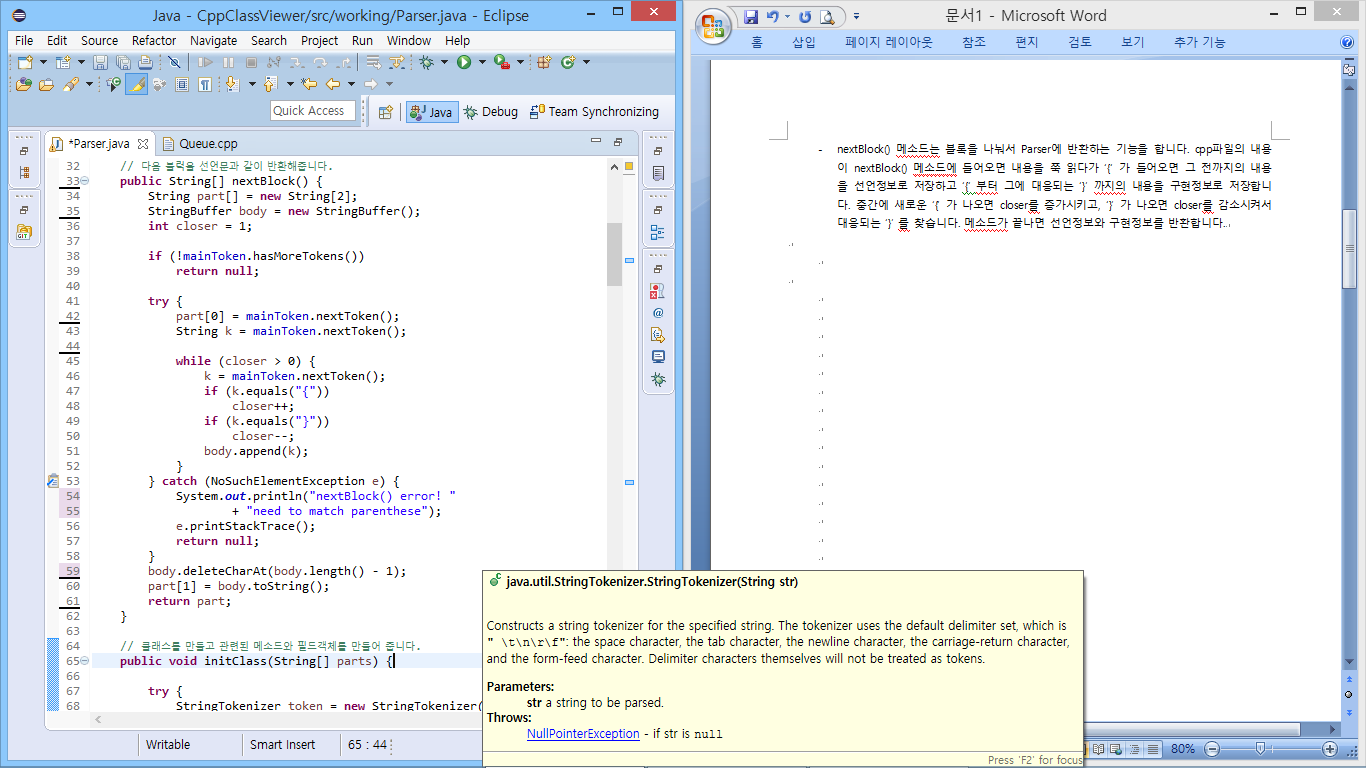
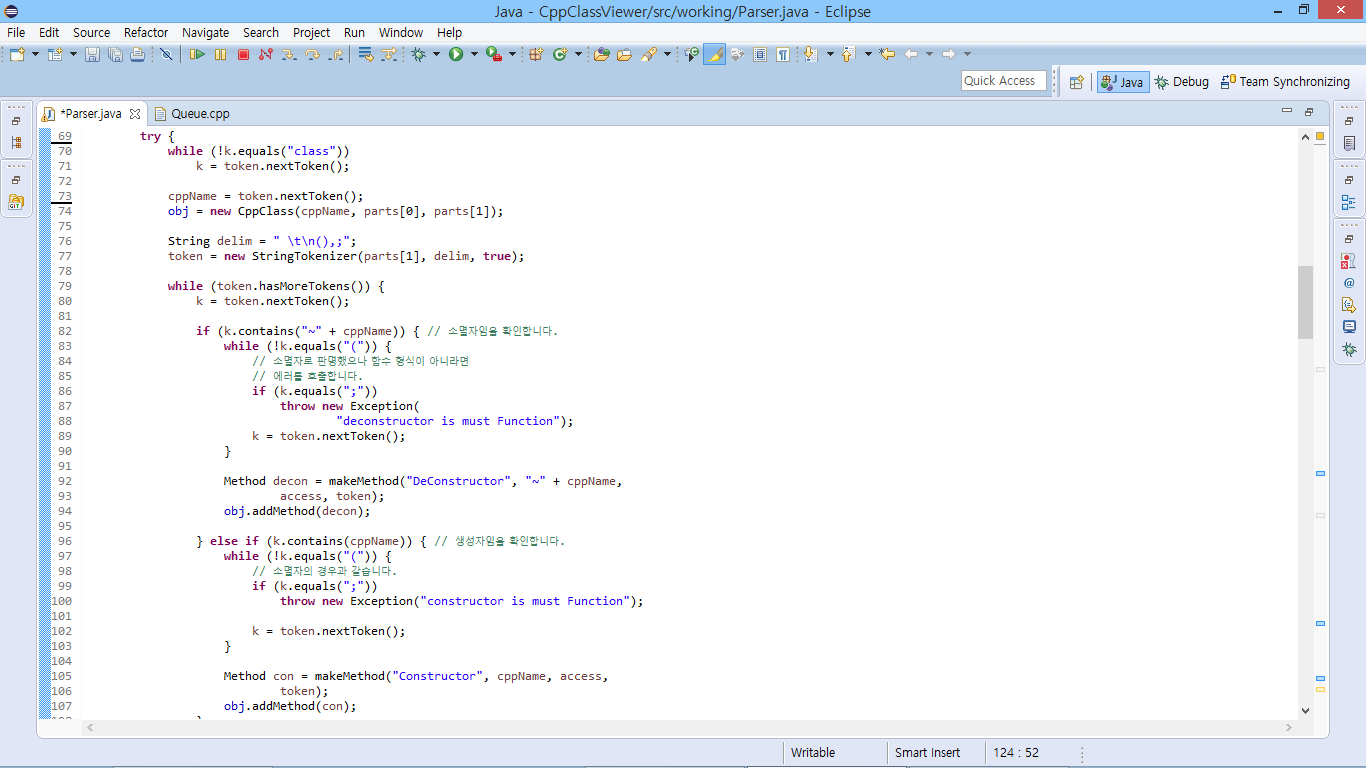
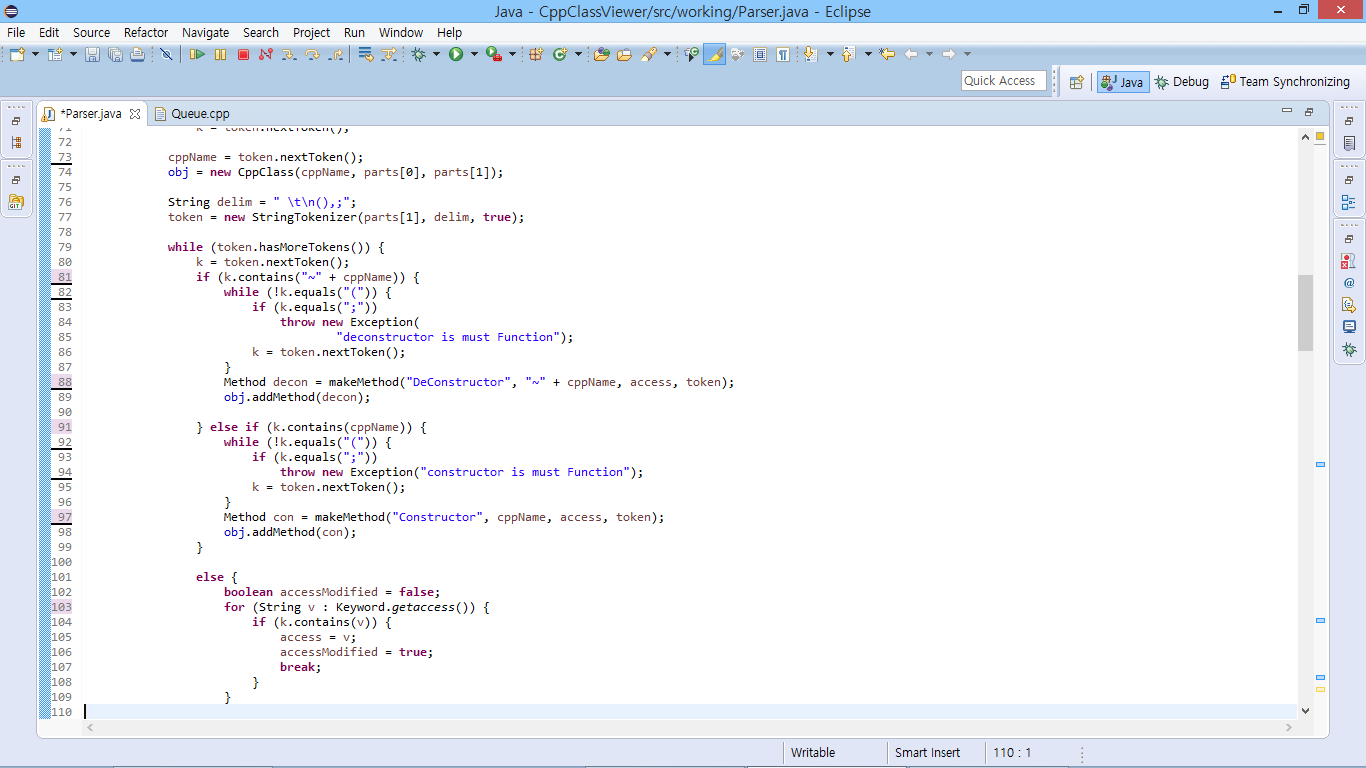
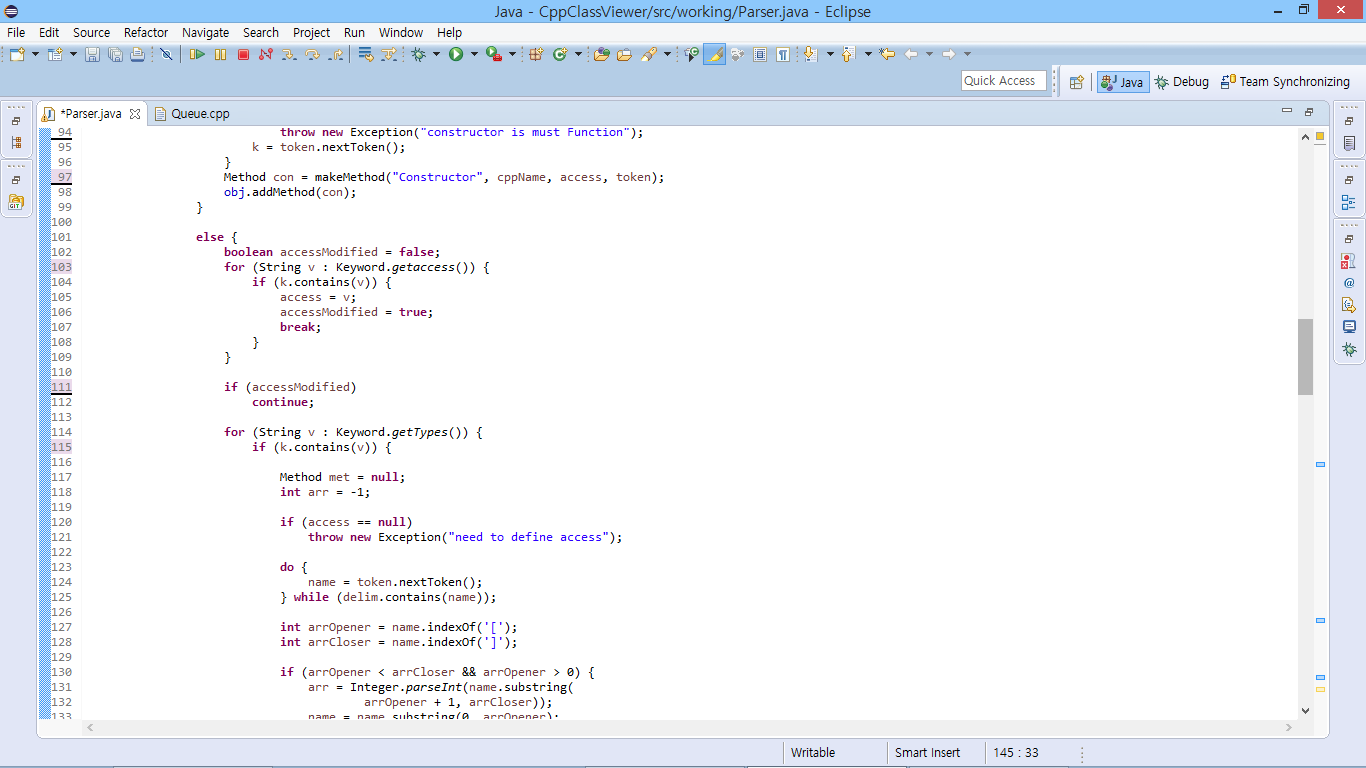
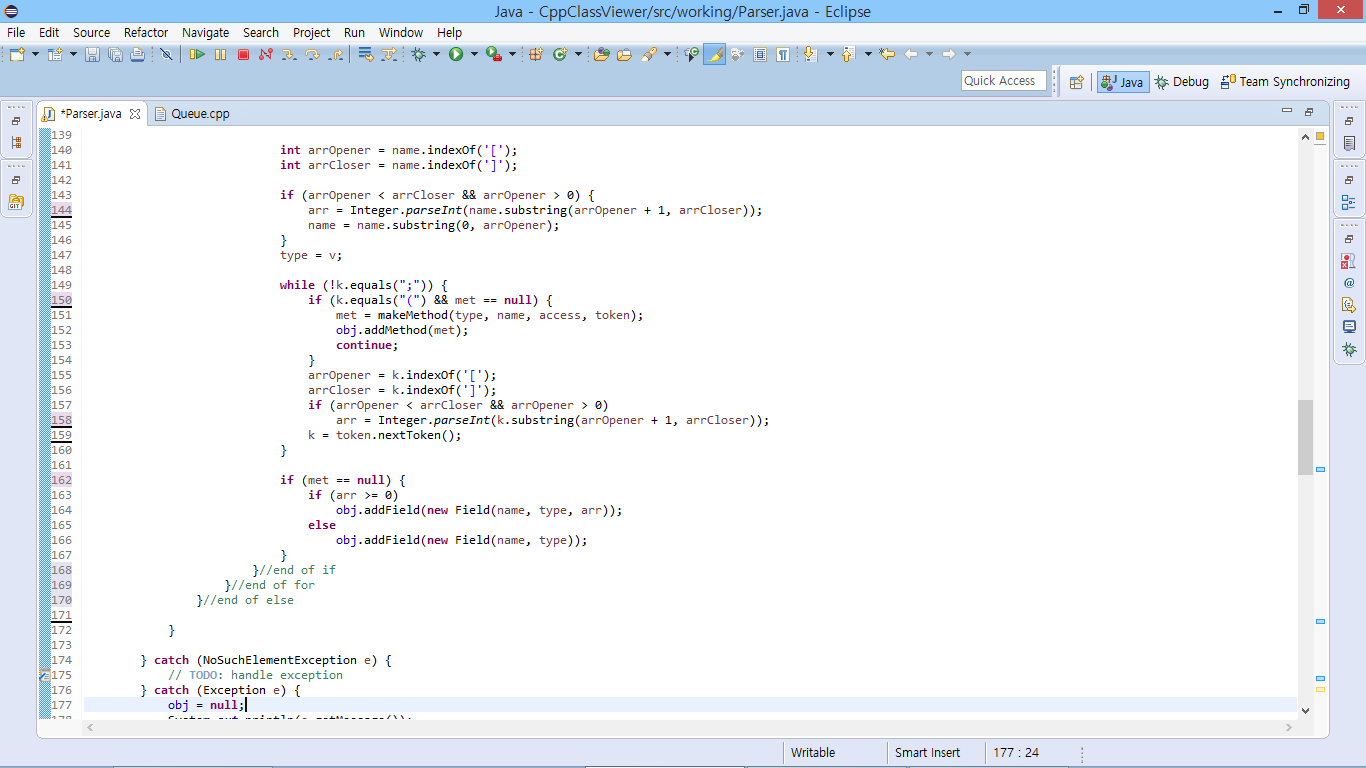
**<Cpp Class Viewer>**

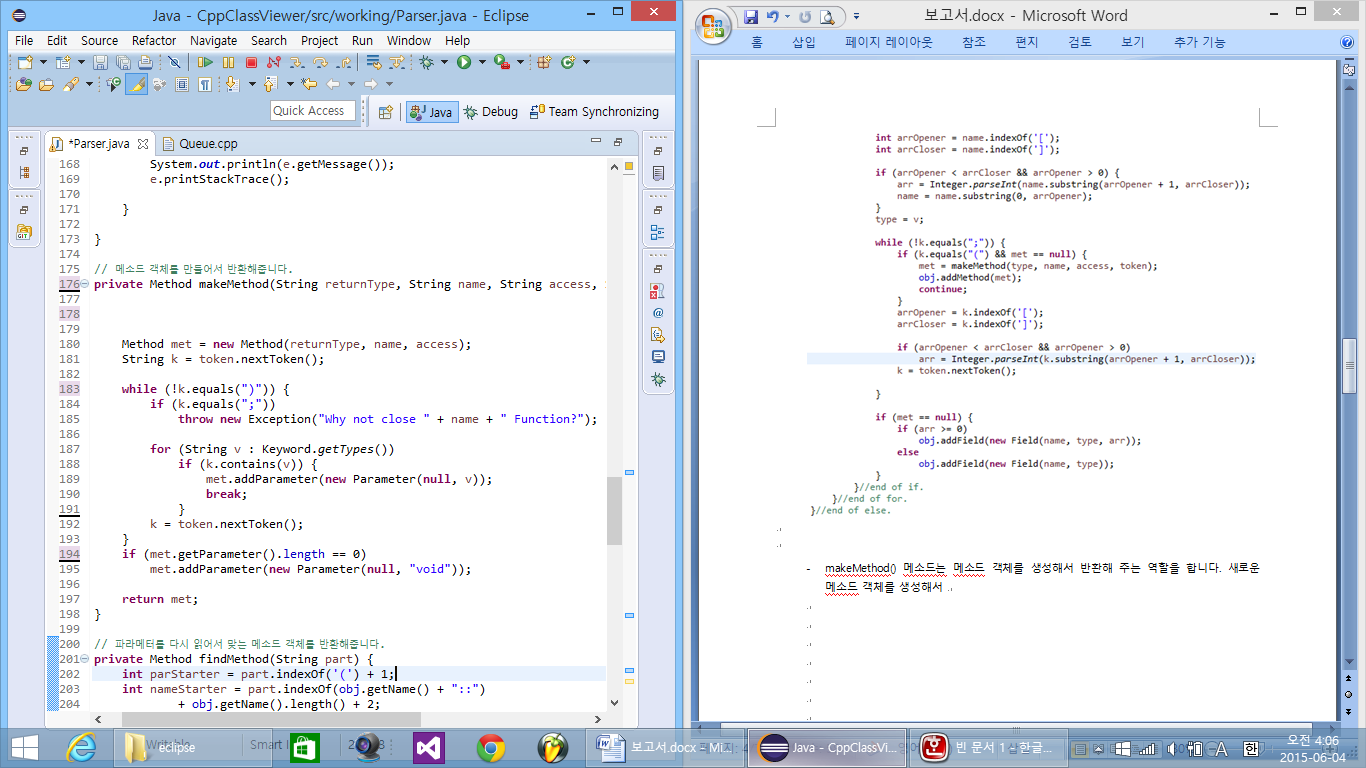
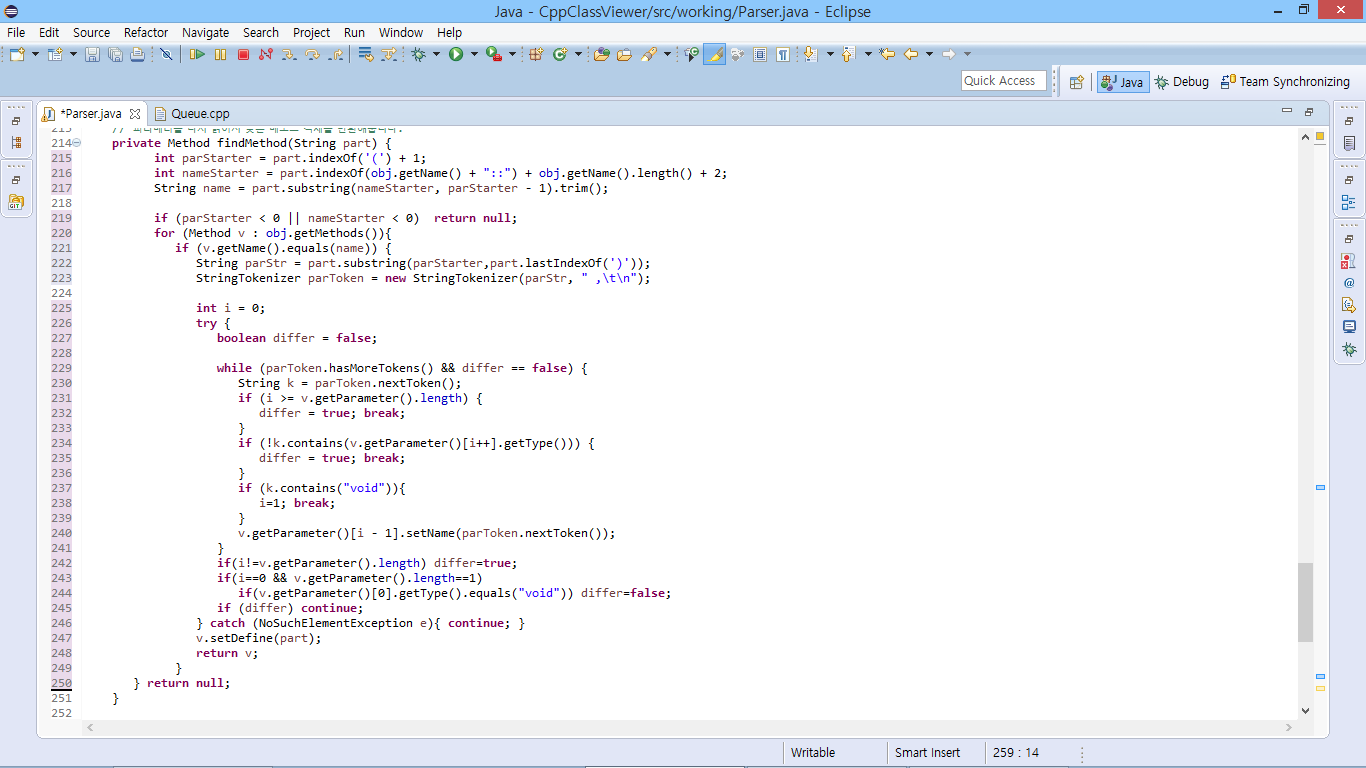
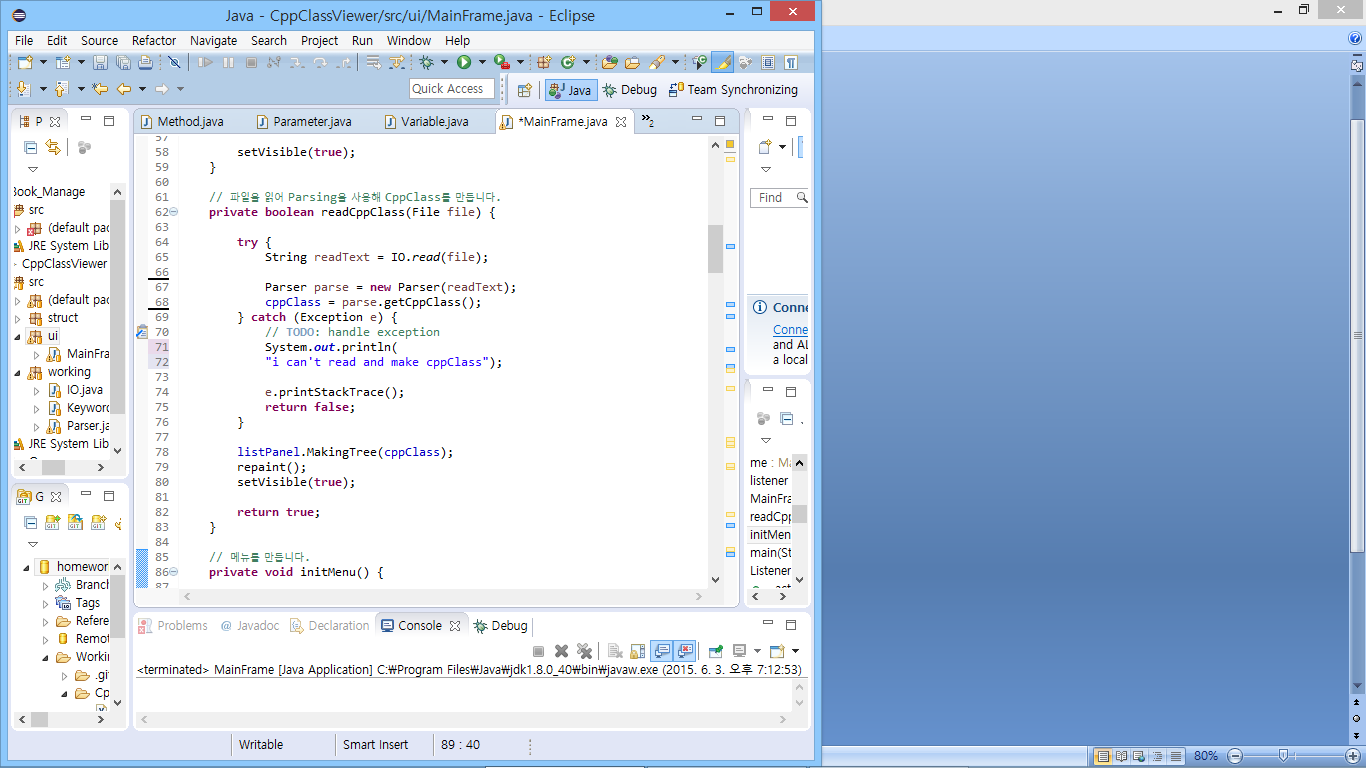
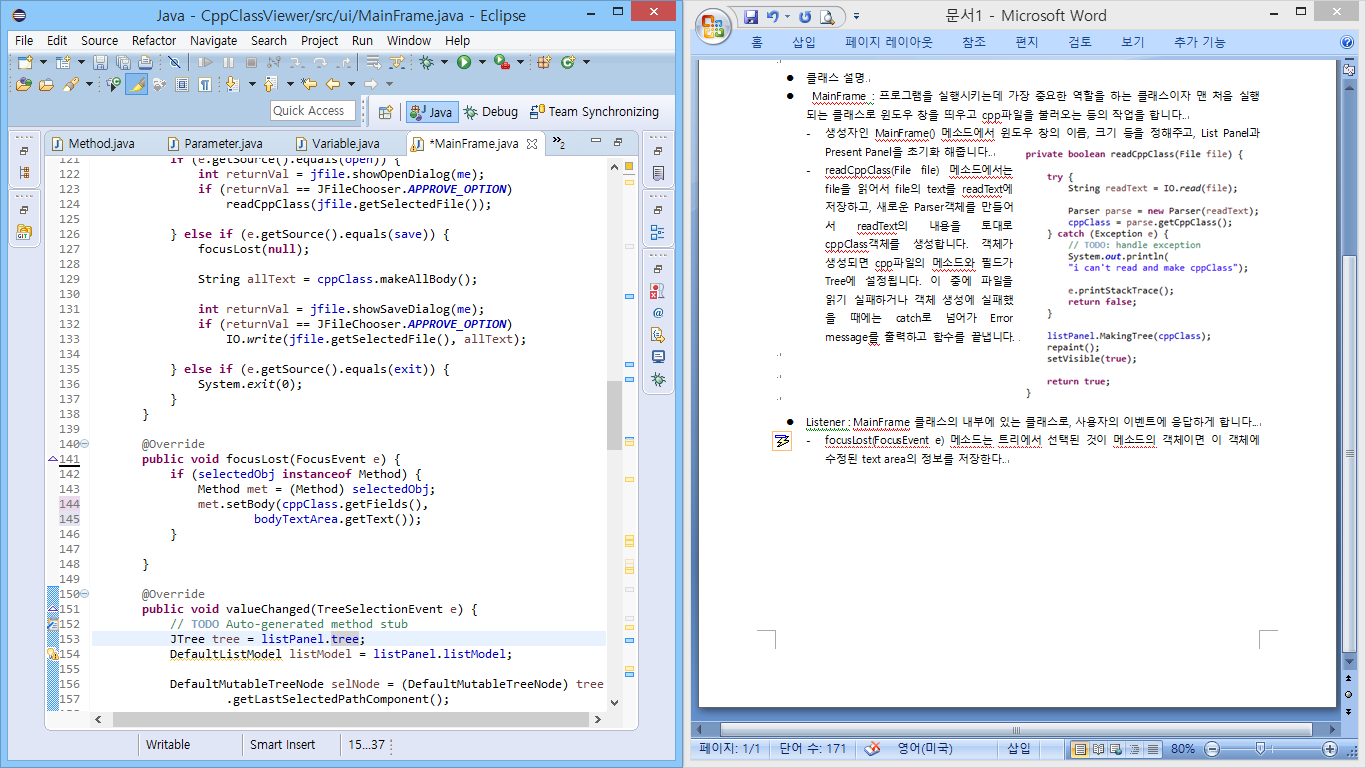
* 소개
* : Cpp Class Viewer는 cpp파일에 사용된 메소드와 필드를 불러와 보여주는 Viewer를 만드는 프로그램입니다. 프로그램을 실행시켰을 때, 창의 왼쪽 상단에 있는 [file]-[open]을 통해 컴퓨터 상에 있는 cpp파일을 불러오고 그 cpp파일에 사용된 메소드와 필드를 트리로 나타냅니다. 메소드를 클릭하면 메소드의 내용을 불러오고 불러온 내용은 수정이 가능하며, 필드를 클릭하면 필드가 사용된 메소드와 그 메소드의 타입, 그 메소드의 접근 지정자를 보여줍니다.
* 클래스 설명 & 실행 방법(1) - working 패키지
* Parser : cpp 파일에 있는 텍스트를 모두 분석해서 cppClass객체를 만들어줍니다.
* 생성자인 Parser() 메소드가 호출되면 자동적으로 cppClass 객체를 생성합니다. nextBlock() 메소드를 호출해서 블록을 나누고 하나씩 가져옵니다. 가장 처음 블록을 읽어서 메소드의 정의와 필드의 선언을 통해 각각의 메소드와 필드의 객체를 만듭니다. 블록을 하나씩 읽으면서 블록마다 구현된 메소드 내용을 해당하는 메소드 객체에 저장합니다.

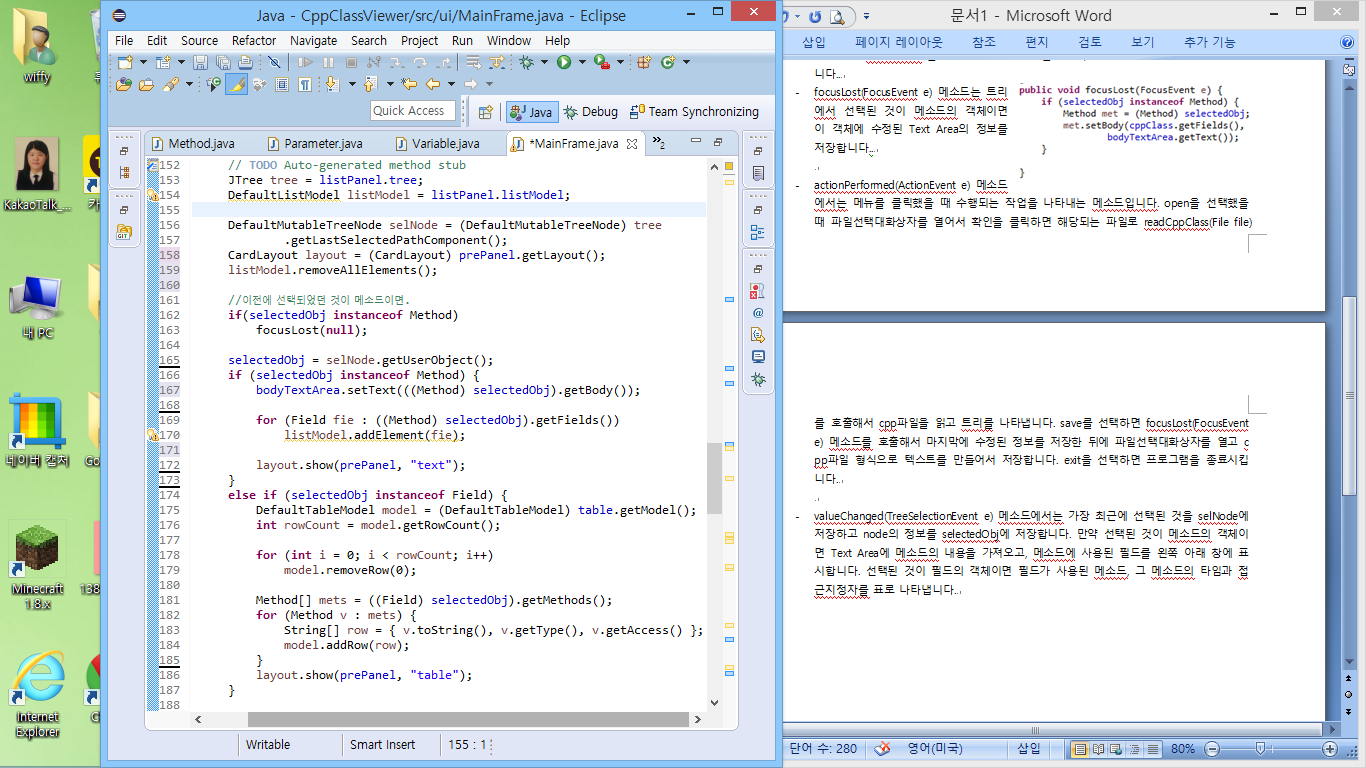


* nextBlock() 메소드는 블록을 나눠서 Parser에 반환하는 기능을 합니다. cpp파일의 내용이 nextBlock() 메소드에 들어오면 내용을 쭉 읽다가 ‘{‘ 가 들어오면 그 전까지의 내용을 선언정보로 part[0]에 저장하고 ‘{‘ 부터 그에 대응되는 ‘}’ 까지의 내용을 구현정보로 part[1]에 저장합니다. 중간에 새로운 ‘{‘ 가 나오면 closer를 증가시키고, ‘}’ 가 나오면 closer를 감소시켜서 대응되는 ‘}’ 를 찾습니다. 메소드가 끝나면 선언정보와 구현정보를 반환합니다.
* initClass() 메소드는 클래스를 만들고 관련된 메소드와 필드의 객체를 생성해줍니다. token을 쭉 읽다가 ‘class’라는 token이 나오면 그 다음에 오는 token을 cppName에 저장하고, 이름에 cppName, 선언정보에 parts[0], 구현정보에 parts[1]의 정보를 가지는 CppClass 객체 obj를 생성합니다. 다시 token을 읽다가 token이 생성자이거나 소멸자일 경우, 함수 형식이 아니면 Error를 호출하고, 함수 형식이면 obj에 메소드를 추가합니다. token이 생성자나 소멸자가 아니면 접근범위를 설정해줍니다. 문장이 타입으로 시작하므로 필드이거나 메소드이고, 타입의 다음에 나오는 것이 이름이므로 단어가 나올 때까지 token을 계속 읽습니다. 만약 이름에 괄호가 존재한다면, 배열의 크기를 읽어서 arr에 저장하고, 문장이 끝날 때까지 token을 읽습니다. 만약 ‘(‘ 가 있다면 메소드일 것이고, 없다면 필드일 것입니다.





* makeMethod() 메소드는 메소드 객체를 생성해서 반환해 주는 역할을 합니다. 새로운 메소드 객체를 생성해서 token으로 메소드의 파라미터를 읽어와 맞게 만들어줍니다. 만약 비어있을 경우 void 파라미터를 만들어줍니다.
* findMethod() 메소드는 파라미터를 읽어서 파라미터 순서가 동일한 메소드 객체를 반환해주는 역할을 합니다. 우선 선언정보를 가지고 있는 문자열에서 파라미터 정보가 시작되는 부분과 클래스 이름 다음에 메소드 이름이 시작하는 부분을 찾고, 둘 중 하나라도 찾지 못하면 null을 반환합니다. 만약 메소드 객체 중에서 이름이 같은 것을 찾으면 파라미터 정보를 가지고 있는 문자열을 생성하고, 이를 이용해 파라미터를 각각 분리합니다. 선언정보의 파라미터가 더 많거나 파라미터 타입이 다르면 다른 메소드라고 표시하고 다음 메소드를 검사합니다. 메소드를 찾으면 선언정보를 메소드에 설정하고, 찾지 못하면 null을 반환합니다..
* 클래스 설명 & 실행 방법 (2) - ui 패키지
* MainFrame : 프로그램을 실행시키는데 가장 중요한 역할을 하는 클래스이자 맨 처음 실행되는 클래스로 윈도우 창을 띄우고 cpp파일을 불러오는 등의 작업을 합니다.
* 생성자인 MainFrame() 메소드에서 윈도우 창의 이름, 크기 등을 정해주고, List Panel과 Present Panel을 초기화 해줍니다.
* readCppClass() 메소드에서는 file을 읽어서 file의 text를 readText에 저장하고, 새로운 Parser객체를 만들어서 readText의 내용을 토대로 cppClass객체를 생성합니다. 객체가 생성되면 cpp파일의 메소드와 필드가 Tree에 설정됩니다. 이 중에 파일을 읽기 실패하거나 객체 생성에 실패했을 때에는 catch로 넘어가 Error message를 출력하고 함수를 끝냅니다.
* InitMenu() 메소드는 메뉴를 설정해주는 메소드로, 메뉴바에서 [file]을 선택했을 때 open, save, exit이 나오게 합니다.
* Listener : MainFrame 클래스의 내부에 있는 클래스로, 사용자의 이벤트에 응답하게 합니다..
* focusLost() 메소드는 트리에서 선택된 것이 메소드의 객체이면 이 객체에 수정된 Text Area의 정보를 저장합니다..
* actionPerformed() 메소드에서는 메뉴를 클릭했을 때 수행되는 작업을 나타내는 메소드입니다. open을 선택했을 때 파일선택대화상자를 열어서 확인을 클릭하면 해당되는 파일로 readCppClass()를 호출해서 cpp파일을 읽고 트리를 나타냅니다. save를 선택하면 focusLost() 메소드를 호출해서 마지막에 수정된 정보를 저장한 뒤에 파일선택대화상자를 열고 cpp파일 형식으로 텍스트를 만들어서 저장합니다. exit을 선택하면 프로그램을 종료시킵니다.
* valueChanged() 메소드에서는 가장 최근에 선택된 것을 selNode에 저장하고 node의 정보를 selectedObj에 저장합니다. 만약 선택된 것이 메소드의 객체이면 Text Area에 메소드의 내용을 가져오고, 메소드에 사용된 필드를 왼쪽 아래 창에 표시합니다. 선택된 것이 필드의 객체이면 필드가 사용된 메소드, 그 메소드의 타임과 접근지정자를 표로 나타냅니다.



* ListPanel : Tree와 List를 가지고 있는 패널입니다.
* PresentPanel : Table과 TextArea를 선택적으로 보여주는 패널입니다.
* 클래스설명 & 실행 방법 (3) - struct 패키지
* Method
* setBody() 메소드는 기본적으로 메소드의 정보를 저장하는 기능을 하지만, 여기에서는 사용되는 필드를 찾아서 필드와 메소드를 서로 저장시키는 기능도 가지고 있습니다.
* toString() 메소드는 메소드 이름과 파라미터 정보를 같이 표현해줍니다.
* CppClass : 하나의 cpp클래스를 표현하는 클래스입니다.
* makeAllBody() 메소드는 CppCass의 메소드와 필드를 읽어서 cpp문법에 맞게 텍스트를 다시 작성해주는 기능을 합니다.