Using Eclipse IDE for Developing and Debugging ByteCraft Pro

Eclipse IDE is a powerful tool for Java development, offering features like code navigation, debugging, and project management. Here's how you can use Eclipse to efficiently develop and debug "ByteCraft Pro" modules:

---

1. Setting Up the Project in Eclipse

Step 1: Create a New Java Project

1. Open Eclipse and select File → New → Java Project.
2. Enter the project name as `ByteCraftPro` and click Finish.

Step 2: Configure Project Structure

1. Add Source Folders:
    - Right-click on the project → New → Source Folder.
    - Create folders like `src`, `test`, etc., for organizing code.
2. Set Up Libraries:
    - Right-click on the project → Build Path → Configure Build Path → Libraries.
    - Add external `.jar` files or specify the `lib` directory for dependencies.

Step 3: Create Packages and Classes

1. Right-click on the `src` folder → New → Package.
2. Create packages like `com.byteminds.bytecraftpro.authentication`, `content`, `payment`, etc.
3. Add Java classes to each package by right-clicking on the package → New → Class.

---

2. Key Features of Eclipse for Development

1. Code Navigation

- Quick Open: Press `Ctrl + Shift + R` to quickly open files.
- Navigate to Definition: Use `Ctrl + Click` on a class or method to navigate to its declaration.
- Outline View: Use the Outline pane to quickly locate methods, variables, and classes in the active file.

2. Code Assistance

- Auto-Completion: Press `Ctrl + Space` for intelligent code suggestions.
- Error Detection: Eclipse highlights syntax and compilation errors in real-time.
- Refactoring: Rename variables, classes, or methods across the project using `Alt + Shift + R`.

---

3. Debugging in Eclipse

Step 1: Add Breakpoints

- Open the Java file where you want to debug.
- Double-click on the left margin to set breakpoints at specific lines.

Step 2: Debug the Application

1. Right-click on the main class (e.g., `Application.java`) → Debug As → Java Application.
2. The Debug perspective will open.

Step 3: Use Debugging Tools

- Variables View: Inspect variable values during runtime.
- Step Into/Over: Use buttons to step into methods or execute the current line (`F5`/`F6`).
- Breakpoints View: Manage all breakpoints in the project.
- Console: View output and logs in the Console window.

---

4. Project Management Features

1. Build Automation

- Eclipse automatically compiles Java files on save.
- Use Project → Clean to rebuild the project when needed.

2. Version Control Integration

- Integrate Git by installing the EGit Plugin (usually pre-installed).
- Use the Git Repositories view to clone repositories, commit changes, and push updates.

3. Task Management

- Use the Tasks View to track TODOs or FIXMEs in the code.

---

5. Tips for Efficient Development

1. Code Templates:
    - Set up reusable code snippets under Preferences → Java → Editor → Templates.
2. Organize Imports:
    - Press `Ctrl + Shift + O` to automatically organize imports.
3. Export Runnable JAR:
    - For deployment, export the project as a runnable JAR:
      File → Export → Java → Runnable JAR File.

---

6. Practical Use Case

Example: Debugging a Login Issue

1. Set a breakpoint in `LoginService.java` at the line where credentials are validated.
2. Run the application in Debug mode.
3. Input incorrect login credentials.
4. Use the Variables view to inspect user data and identify the issue.