Coding Standards

Presented By :
Sangeeta Joshi

Agenda

- Coding Standards – The Need
- Coverage of Coding Standards
- Benefits
- Challenges In Implementation

Coding Standards – The Need

- Readability
- Maintainability
- Compatibility
- Standardization of code
- Helps in generating documentation.

Coverage of Coding Standards

- Naming Conventions
- File Naming and Organization
- Formatting and Indentation
- Comments and Documentation
- Good Programming Practices
- Exception Handling

Naming Conventions

- Length of identifiers
- Letter case and numerals
- Multiple-word identifiers
- Meaningful methods and variable names

File Naming and Organization

- Filenames
  - Name the code files based on the purpose that it serves in the system
  - If a file does more than one logical task then split the files and name them accordingly
- Folders
  - Logically organize files within appropriate folders
  - Use 2 level folder hierarchies.
  - You can have up to 10 folders in the root folder
  - Each folder can have up to 5 sub folders

Formatting and Indentation

- Readability
- Identification of code blocks
- Line Length
- Cross OS Compatibility

Formatting and Indentation ..

- if (hours< 24 && minutes< 60 && seconds< 60) {return true ;} else {return false ;}

Or

- if (hours < 24 && minutes < 60 && seconds < 60)
  {
  return true;
  }
  else
  {
  return false;
  }

Comments and Documentation

- Helps Following Program Flow
- Helps Skip *Unwanted Code/ out of scope Code*
- Comments should be at all levels of the code
- Helps in creating automated docs (ex: Javadoc)

Best Industry Practices

1. **Refactoring Exercise in Eclipse**
2. Class Naming Convention
3. Method and Variable Naming Convention
4. Comments
5. Separation Of code

6. Separation of packages
7. Formatting Code and Organizing Imports
8. Documentation
9. Avoid Repetition of Code
10. Using tools for best practices like PMD, Sonarlint

Best Industry Practices Demo On Below Pointers

1. Using Naming Conventions
2. Ordering Class Members by Scopes
3. Class Members should be private
4. Using Underscores in Numeric Literals
5. Avoid Empty Catch Blocks
6. Using StringBuilder or StringBuffer instead of String Concatenation
7. Using Enums or Constant Class instead of Constant Interface
8. Avoid Redundant Initialization (0-false-null)
9. Using Interface References to Collections
10. Avoid using for loops with indexes

Good Programming Practices

- Avoid writing long methods
- Method names should be meaningful
- A method should do only 'one job'
- No hardcoding
- Care while string comparisons
- Avoid writing large code files
- Good exception handling and logging

Exception Handling

- Always catch specific exceptions and not generic
- Give friendly message to user, but save the details to a log
- Never do a *'catch exception and do nothing'*
- Use the mechanism to write the cleanup code

Benefits of Coding Standards

- Code Integration
- Team Member Integration
- Maintenance
- Uniform Problem Solving
- Minimizes Communication
- Minimizes Performance Pitfalls
- Saves Money Due to Less Man Hours

Challenges Using Coding Standards

- Customization of coding Standards
- Standardization across the team/s (Onsite/Offshore)
- Imbibing faith in coding standards in developers
- Verification of the standards
- Handling Legacy Code

Questions ??

Thank You