ONE PAGE SUMMARY

IT & Software › Git & GitHub › Conflict Resolution

Introduction to Conflict Resolution

Mastering Conflict Resolution in Git for Smooth Collaboration

DEFINITION

> Conflict resolution in Git is the process of addressing and merging conflicting changes in code. It ensures that multiple contributors can work together without overwriting each other's work.

KEY CONCEPTS

Merge Conflicts

Merge conflicts occur when two branches have competing changes to the same line of code.

Conflict Markers

Conflict markers are special symbols in code that indicate where conflicts exist and need resolution.

Manual Resolution

Manual resolution involves editing the conflicting code directly to choose which changes to keep.

Using Tools

Many IDEs and Git tools provide visual interfaces to help resolve conflicts more easily.

Testing After Resolution

Always test your code after resolving conflicts to ensure functionality remains intact.

EXAMPLES

→ Two developers edit the same function in different branches, causing a conflict.

→ A developer resolves a conflict using a visual merge tool.

→ After merging, a developer runs tests to confirm no errors were introduced.

MEMORY TIPS

★ Remember 'M' for Merge and 'C' for Conflict to link the concepts.

★ Think of 'Conflict Markers' as road signs indicating where to stop and check.

★ Use the acronym 'RMT' for Resolve, Merge, Test to recall the steps.

COMMON MISTAKES

✗ Ignoring conflict markers and merging without resolution.

✗ Failing to test code after resolving conflicts.

✗ Not communicating with team members about changes made during resolution.

QUICK RECAP

Conflict resolution in Git is crucial for collaborative coding. It involves identifying, resolving, and testing conflicts to ensure smooth integration of changes. Always communicate with your team and use tools to aid the process.