

① DECLARATIVE PARALLEL TEMPLATE (COPY THIS ANYTIME)

```
pipeline {  
    agent any  
  
    stages {  
  
        stage('Initial Stage') {  
            steps {  
                // example: git clone, setup, login  
                sh 'echo initial work'  
            }  
        }  
  
        stage('Parallel Work') {  
            parallel {  
  
                stage('Task A') {  
                    steps {  
                        // commands for task A  
                        sh 'echo task A'  
                    }  
                }  
  
                stage('Task B') {  
                    steps {  
                        // commands for task B  
                    }  
                }  
            }  
        }  
    }  
}
```

```

        sh 'echo task B'

    }

}

}

stage('Final Stage') {
    steps {
        // deploy, upload, notify
        sh 'echo final stage'
    }
}
}
}
}

```

How YOU fill this in future

Replace:

Task A → sonar scan
 Task B → tests

Example:

```
sh 'mvn sonar:sonar'
sh 'npm test'
```

Interview one-liner:

 “Declarative parallel uses parallel block inside a stage.”

SCRIPTED PARALLEL TEMPLATE (COPY THIS ANYTIME)

Scripted syntax looks different — remember this.

```
node {

    stage('Initial Stage') {
        sh 'echo initial work'
    }

    stage('Parallel Work') {
        parallel(
            "Task A": {
                // commands for task A
                sh 'echo task A'
            },
            "Task B": {
                // commands for task B
                sh 'echo task B'
            }
        )
    }

    stage('Final Stage') {
        sh 'echo final stage'
    }
}
```

Key Difference (VERY IMPORTANT)

Declarative:

stage → parallel → stage → steps

Scripted:

```
stage → parallel( "branch": { code } )
```

No `steps` {} in scripted.

Interview one-liner:

👉 “Scripted parallel uses map-style closures instead of stage blocks.”

⚠ CAUTIONS WHEN YOU WRITE YOUR OWN

✓ Only parallelize independent tasks

Good:

```
Tests + Sonar
```

Bad:

```
Build + Deploy
```

✓ Always keep setup BEFORE parallel

Example:

```
Git clone  
Login  
Environment setup
```

✓ Deployment usually AFTER parallel

Because deploy needs finished artifact.

✓ Give meaningful stage names

Don't use:

```
stage('parallel')
```

Use:

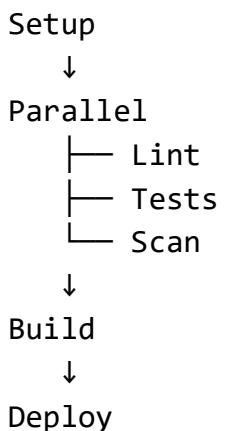
```
stage('Quality Checks')
```

❖ HOW TO DESIGN FUTURE PARALLEL JOBS (Your Thinking Formula)

Before writing code, ask:

What tasks can run at same time?

Draw:



Then convert drawing → template.

⌚ FINAL CHEAT SHEET (MEMORIZE THIS)

Declarative:

```
stage {  
    parallel {  
        stage { steps {} }  
        stage { steps {} }  
    }  
}
```

Scripted:

```
stage {  
    parallel(  
        "A": { code },  
        "B": { code }  
    )  
}
```