**Documentation Types**

|  |  |  |
| --- | --- | --- |
| **Document** | **Definition** | **Document’s elements** |
| API Reference | The API Reference provides detailed information for each class and method of an API, including examples of how developers can use them, and the constraints that the API allows. | (e.g., Class Summary, Field Summary, Constructor Summary, Method Summary, Field Detail, Constructor Detail, Method Detail) |
| Code Comment | Code Comments summarize a piece of code or explain the programmer's intent. | (e.g., Functionality & Behaviour, Parameters, Purpose and Rationale, References) |
| Contribution Guideline | Contribution Guideline explains how developers should contribute to the project (e.g., format code, test fixes, and submit patches). For that, usually open source projects place a CONTRIBUTING file in the root directory. | (e.g., Coding Style, Testing, Licensing, Pull-Request Checklist) |
| Deployment Guide | Deployment Guide describes the steps that are necessary to deploy the software in a new environment (usually the production environment). | (e.g., Prerequisites, Installation Steps, Configuration, Troubleshooting, Security Guidelines) |
| FAQ | FAQ provides a list of Q&A which addresses the most frequent questions/issues developers face when using the project (e.g., Common error messages, Is Tomcat faster than serving static HTML pages than Apache httpd?) | (e.g., Table of Content, Common Issues, Basic Usage Questions, How-to Questions, General Questions, License-Related Questions) |
| How-To / Tutorial | Tutorials are usually short documents which guide the reader with a set of step-by-step instructions to complete a specific task or to learn an application's feature. | (e.g., Prerequisites, Examples, Screenshots, Unrecommended Usages, Best Practices) |
| Installation Guide | Installation Guide provides detailed procedures on installation, upgrade, or uninstallation of the software. It usually includes information such as System Requirements, Quick Start Configuration, Set-Up Configurations. | (e.g., Prerequisites, Installation Steps, Configuration, Troubleshooting, Security Guidelines) |
| Introduction / Getting Started Document | Introduction / Getting Started Document is a guide designed to quickly walk the reader through the core concepts and basics of setting up a software package. | (e.g., Quick Start Installation, Minimal Working Project) |
| Migration Guide | Migration Guide (or Transition Guide) is a set of step-by-step instructions that helps users to migrate their data to another (usually newer) version of a software or to integrate a newer version of a framework/library in their codebase. | (e.g., Back-up/Restore Guidelines, Deprecated Functions, Manual Migration Steps) |
| Release Note / Change Log | Release Note / Change Log provides a brief overview of the changes in a new release/version of a product with a list of what is new in it, e.g., bug fixes and new features. | (e.g., Resolved Issues, New Features, Upgrade Instructions) |
| User Manual | User guides (or Software Manual and User Manual) presents an overview of an application’s features and gives step-by-step instructions for completing a variety of tasks. | (e.g., Table of Content, Technical Description, Screenshots, Main Options, Examples, Contact Details, Troubleshooting) |
| Video Tutorials | A tutorial (i.e., a step-by-step instruction to complete a specific task or to learn a feature) in the video format. | (e.g., Playlist, Time Links, Audio Narration, Captions/Subtitles) |
| Community Knowledge | Community Knowledge refers to the information available on sources such as mailing list, Q&A websites (e.g., Stack Overflow), and forums. | (e.g., Question & Answers, Forums Discussion, Mailing-Lists Discussion, Chat Rooms, Social Media, News/RSS Feeds) |

**Software Development Activities**

|  |  |
| --- | --- |
| **Activity** | **Definition** |
| Requirements Engineering | Activities related to the process of defining, documenting and maintaining requirements, including requirements elicitation, analysis, specification, and validation. |
| Software Structure and Architecture Design | The activity of designing the high-level structure and architecture of a software system, i.e., the set of structures needed to reason about the system, which comprises software elements, relations among them, and properties of both. |
| User Interface Design | The process of designing a user interface of a software system or a computerized device with a focus on look or style. |
| Database Design | Database design is the organization of data according to a database model. During this process one determines what type of information must be stored and how the data elements interrelate. |
| Quality Attributes Analysis and Evaluation | Analyzing and assessing a software system with respect to a variety of software quality attributes (e.g., modifiability, usability, maintainability, reliability, security). |
| Software Programming | Programming is the process of designing and building an executable computer program to accomplish a specific computing task. It involves tasks such as analysis, generating algorithms, and the implementation of algorithms in a chosen programming language (commonly referred to as coding). |
| Software Debugging | Debugging is the process of locating and removing computer program bugs, errors or abnormalities, which is methodically handled by software programmers via debugging tools. |
| Code Refactoring | Code refactoring is the process of restructuring existing computer code without changing its external behavior. Refactoring is intended to improve nonfunctional attributes of the software. |
| Program Comprehension | Program Comprehension is the process of developing mental models of a software system intended architecture, meaning, and behavior. It aims to recover high-level information about a system including its structure, functionality, dynamic behavior, and rationale. |
| Reverse Engineering and Design Recovery | The process of analyzing a subject system to identify the system’s components and their interrelationships, and to create representations of the system in another form or at a higher level of abstraction. |
| Software/Data Migration | Software migration is the practice of transferring data, accounts, and functionality from one operating environment to another. It could also refer to times when users are migrating the same software from one piece of computer hardware to another, or changing both software and hardware simultaneously. |
| Release Management | Release management is the process of managing, planning, scheduling and controlling a software build through different stages and environments; including testing and deploying software releases. |
| Legal Aspects (Copyright, License, etc.) | Legal issues related to software engineering professional practice notably include matters pertaining to standards, trademarks, patents, copyrights, trade secrets, professional liability, legal requirements, trade compliance, and cybercrime. |
| Software Testing/Quality Assurance | Quality Assurance is the implementation of policies and procedures intended to prevent defects from reaching customers, including bug/issue reporting and software testing, an investigation conducted to provide stakeholders with information about the quality of the software product or service under test. |
| Learning a New Technology/Framework | The process of gaining knowledge of or skill in a new technology/framework by study or experience. |