Java基本用語集

1. 開発関係の用語
2. ウォーターフォール開発

システム開発手法の一つ。開発工程を「要件定義」→「システム設計」→「プログラミング」→「テスト」の順に滝のように進めていく。

1. アジャイル開発

ウォーターフォール開発では、開発期間が長く、利用者が完成品を見るまでに時間がかかるため、アジャイル開発が使われる。これは「短期間で動作するアプリケーションを開発する」作業を反復させて、クライアントの要求を常に取り入れながら、段階的にシステムを完成させる。

1. マイグレーション

ソフトウェアやハードウェア、システム、データ、開発言語などを別のプラットフォームに移行したり、新しいシステムに切り替えること。

1. デグレーション

プログラムの不具合を修正したときに、新たな不具合が出てくること。

1. デプロイ

開発したアプリケーション（機能やサービス）をサーバー上に展開・配置して利用できるようにすること。

1. コンフリクト

Gitにおいて変更内容をMergeにより統合する前に、同じファイルの同じ部分が変更されていると、変更したもの同士が衝突してコンフリクトという状態になる。解決方法は2つあり、1つ目はコード上のコンフリクトを解決して、git commitによってマージコミットをコミットするか。マージを中止することである。

1. 環境
   * 開発環境（ローカル）

個人のコンピューター内に構築された環境のこと。

* + 検証環境（ステージング）

開発環境で作られたシステムをテストするための工程。開発環境と分けることで、別の開発環境と並行してテストできるようになる。ここでは、単体テスト、結合テストなどが行われる。

* + 本番環境

出来上がったシステムやサービスが実際に動く環境（コンピュータとかソフト）のこと。

1. 設計書 / 仕様書
   * 要件定義

利用者のニーズを整理して、システムの仕様を明確化し、システム化の範囲とその機能を定義する。クライアントの要望と、それをどのように叶えるのかを文章としてまとめたもの。クライアントの要望を聞くために、通常はヒアリング調査をする。最終的には要望をまとめたものをクライアントに提出する、これを要件定義書という。

* + 基本設計書

ソフトウェアの動きを外から見た際にどういう動きをするのかを決めたもの。（Whatを決めたもの）基本設計書を作る際には要件定義書を参考にする。それに基づいて外から見た動きを決定する。　設計する内容としては、業務フロー、機能一覧表、ネットワーク構成図、テーブル定義、ER図、画面レイアウト、帳票レイアウトなど。クライアントも一緒に確認することが多いのでクライアントにもわかるように作る。

* + 詳細設計書

基本設計とは異なり、詳細設計書は社内で開発向けに作るもの。基本設計書をどのように実現するかを説明した資料。

1. テスト
   * 単体テスト（UT）

プログラムを構成する比較的小さな単位が個々の機能を正しく果たしているかを検証するテスト。関数やメソッドが単体テストの単位となる。コード作成時などの早い段階で開発者によって実施されることが多いのが特徴。

* + 結合テスト（CT/ IT）

単体テストを実施した後に実行されるテスト。テストタイプの目的を柔軟に変更できることから、プロジェクト中盤から、終盤にかけて行われる。単体テストで無事に作動したプログラムを組み合わせて実際に作動するか確認する。データの受け渡しが正常に行われているかを検証する。

* + 総合テスト（ST）

構築したシステムが予定通りの機能を満たしているか、機能は性能通りか、を検証する。結合テストの後に実施する。本番とほぼ同じ環境でテストを実施する。

1. エビデンス / 証跡

システム開発などのテスト工程などで、システムが正しく動作していることや不具合が生じたことを示す証拠となる書類やデータをこのように呼ぶ。画面のキャプチャやログデータなど。

1. データパッチ

コンピューターにおいてプログラムの一部分を更新して、バグ修正や機能変更を行うためのデータのこと。修正プログラムや、アップデートとも呼ばれる。パッチを当てるというように使われる。

1. Access

Microsoft製のデータベース管理ソフトのこと、

1. マクロ
   * VBA

Visual Basic for Applicationsの略でMicrosoft Officeに含まれるアプリケーションソフトの拡張機能である。簡単なプログラムを記述して実行することで複雑な処理の自動化などを行うことができるもの。繰り返し行われる定型的な作業や複雑な処理を自動的に実行することが出来る。

1. GAS (Google Apps Script)

Googleが開発・提供しているプログラミング言語。これを用いることでGoogleのツール機能の拡張（メールの自動送信）やGoogleが提供しているサービスとの連携が容易にできる。

1. API（プログラムからアプリケーションを操作するための仕組み）
   * WEB API

HTTPを介して呼び出すAPI

* + REST API (Representational State Transfer)

RestとはAPIの設計原則の一つ。特徴としては、URLが名詞形であること、リソース操作（作成・閲覧・変更・削除）をHTTPのメソッドにより指定できること。レスポンス形式がJsonもしくはXMLであること。

* + SOAP (Simple Object Access Protocol)

異なるコンピュータ上で動作するプログラム同士がネットワークを通じてメッセージを伝え合い、連携して動作するための通信プロトコルのひとつ。メッセージの記述にXMLを、データ転送にHTTPを用い、Webサービスの提供や利用に適している。

* Rest APIとSoap APIの違い

RESTは不特定多数を対象にした、入力パラメータが少ない情報配信や検索サービス等での利用に向いている一方、SOAPは複雑な入力を必要としたり、入出力に対して、チェックを必要とするようなサービス等での利用に向いている。

1. Ajax
   * XML

文章の見た目や構造を記述するためのマークアップ言語の一種。主にデータのやり取りや管理を簡単にする目的で使われ、記述形式が分かりやすいという特徴がある。

* + JSON

JavaScript Object Notationの略であり、XMLと同様のテキストベースのデータフォーマット。XMLよりも簡潔に構成されているので、人間が理解しやすい。

* + ステータスコード

HTTPステータスコードと呼ばれるHTMLを表示する前にブラウザが受信する3桁の番号。100～510までの間に割り当てられている。番号はHTMLを受信する前に受信する。

* + クロスオリジン(Cross-Origin) / CORS (Cross-Origin Resource Sharing)

オリジンとはスキーム（http://）、ホスト（www.exsampel.com）、ポート（80）の組み合わせ。

CORSとはあるオリジンで動いているWebアプリケーションに対して、別のオリジンのサーバーへのアクセスをオリジン間HTTPリクエストによって許可できる仕組み。

1. バリデーション

入力場所、書式設定、データの形式などが、既定のマニュアル通りであるかを確認すること。

1. データ連携
   * Daemon

Unix系のOS（LinuxとかMacとか）における常駐プログラムの呼び名。デーモンはバックグラウンドプロセスとして働くプログラムと説明されることがある。デーモンの名前には、最後にｄを付けるのが慣例となっている。

* + CRON

Linuxサーバーの管理で、大切な設定方法の一つ。何をいつ実行するかを決めて、そのルールを書き込むこと。（定期的に実行する）これを設定しておくと、プログラムの動作を予約出来たり、定期的に実行するプログラムを調整したり、効率よくサーバーの管理がしやすくなる。（例：定期的なファイル転送を自動化して、Webサイトを楽に更新する。Webサイトを定期的にバックアップするなど…）

* + バッチ

決められたタイミングで処理を一括して行うこと。主に利用されるものとして、日次処理など。指定された時間に自動で実行したり、ユーザの入力を必要とせずに動作する、終了時に実行結果が分かるようにするなど。

* + ログ

実行時の情報を文字列として出力したもの。プログラム実行時の変数の内容やメソッドの呼び出し順などの内容等はデバッガ等を使用しなければ確認できない。これにより文字列形式でエラー内容やファイルに関する出力する機能やその内容のこと。

* + CSV

Comma Separated Valueの略。データのやり取りで使われていて、互換性が高いファイル。カンマで区切ったテキストデータおよびテキストファイル。

* + TSV

Tab Separated Valueの略。CSVファイルのカンマがタブになっただけ。

1. プルリクエスト（マージリクエスト）

プルリクエストは、Githubのリポジトリにpushにした変更や追加機能をほかの人のリポジトリに取り込んでもらうための要求を出す機能。これが送信されると、相手のリポジトリの管理者は送られてきたプルリクエストの内容やソースコードの変更などを確認できる。

1. チケット管理ツール（タスク管理ツール）

プロジェクト毎に発生する各種タスクを発行して、制作者に対して仕事を依頼するためのもの。

1. Asana

タスク管理がメインの高機能プロジェクト管理アプリ。プロジェクトの進行状況が可視化でき、100以上のアプリと連携が取れることが特徴。

1. Backlog

プロジェクトで発生するタスクなどの進捗状況を一括管理できるプロジェクト管理アプリ。多くのエンジニアに使われるツール。進捗を視覚的に把握でき、タスクに関するコミュニケーションが出来る、またGitなどを利用してソースコードを管理できるためBacklog上でコードレビューが出来る。

1. Jooto

かんばん方式のタスク管理ツール。かんばん方式の原理法則とは、見える化、整流化（シングルタスク）、平準化（負荷の平準化、能力の平準化）、学習（ＰＤＣＡサイクル）の４つ。かんばん方式はかんばんをタスクカードの置き換えたもの。Jootoはシンプルな機能（メモからプロジェクト管理まで幅広く利用できる）とカラフルなインターフェイスで使いやすい。

1. Redmine

オープンソースソフトウェアのプロジェクト管理ツールで、チームでシステム開発を進める際によく使われている。チケットという名称でタスク管理を行う。Textile記法でwikiを作成することが出来る。専門的な知識が必要。（自社サーバーへのインストールなど）

1. Trello

かんばん方式のタスク管理が出来るプロジェクト管理ツール。付箋を貼る感覚で、タスクを追加することが出来、タスクをドラッグ＆ドロップでカードを動かしてタスクのステータスを管理できるから視覚的に把握しやすくなる。デメリットとしては、チーム全体の進捗が把握しにくいこと、

1. バージョン管理

変更履歴を管理すること。具体的にはソフトウェアのソースコードを書き足したり、変更したりする過程を記録してくこと。

1. 集中型

リポジトリをサーバーに集中させて配置する。リポジトリは1つしか存在しない。すべてのデータがサーバーにあるので管理はしやすいが、サーバーに接続できないと最新のソースコードを取得することができない。

1. 分散型

Gitはこの分散型という方法をとっている。ユーザーごとにリポジトリをFork（Github側にある特定のリポジトリを自分のアカウント以下のリポジトリに複製する）すること。複数のリポジトリがあるのが特徴であり、手元の開発環境にもリポジトリがあるのでリモートリポジトリに接続できなくても開発することが出来る。

1. GitHub

様々な人とコードを共有するための場所を提供しているGitリポジトリのホスティングサービス。

GitHubで公開されているソフトウェアのソースコードはすべてGitで管理されており、Gitリポジトリをインターネット上に公開しているのがGitHubである。

1. Backlog

チームが協力しながら作業を進めるためのコラボレーション型プロジェクト管理ツールのこと。企画・マーケティング・総務の各種業務から開発のプロジェクトまで様々なタスク管理に活用できる。

これを用いることで誰が、何を、いつまでに作業するのかがはっきりし、状況が分かりやすくなる。

1. SourceTree

Gitの操作が出来る無料のデスクトップアプリケーションのこと。Gitを簡単に使えるようにしたツール。GUI形式でGitを使うことが出来る。

1. Tortoise SVN (Sub VersioN )

アパッチという団体が開発しているファイルのバージョン管理をしてくれる「バージョン管理システム」の１つ。テキストファイルをはじめ、Excel, Wordなどのファイルなども管理することが出来る。コマンドプロンプト画面で用いらなければならない。

1. Git Flow→リリース中心の開発スタイル

チームでGitを使う際に役立つ、運用ルールをまとめたツール。Gitのデメリットである、開発スタート時に運用ルールを定めないと、プロジェクトがまとまらないというのを解決してくれる。

* Master

プロダクトとしてリリースするためのブランチ。リリースしたらタグ付けする。

* Release

プロダクトリリースの準備。ブランチ元がdevelopであり、リリース前の微調整を行うブランチである。小さなバグ修正が行われる。

* Develop

開発を行うブランチ。

* Feature

ブランチ元がdevelopであり、追加機能などを作成するブランチ。

1. FW / ライブラリ
2. FW（FirmWare）ハードウェアを動かすためのソフトウェア
   * Spring Framework

Javaの開発環境で使用されるオープンソースのフレームワーク（アプリケーション開発を行う際に必要となる機能や基本的な骨組みをまとめたもの）Webシステムやクラウドアプリケーションやモバイルシステムなど様々な開発に適している。保守性が高い

* + Struts

Javaのフレームワークで高い拡張性から幅広いジャンルのアプリやソフトが開発できる。Strutsの中で推奨されているものがMVCシステムデザイン(Model View Controller)であり、これを採用することで分業がしやすくなり、機能ごとに専門的に開発をすることが出来る。また、Strutsでは独自のタグライブラリが用意されているため、これを利用して開発ができることで質の良いアプリの開発が出来る。

* + JSF (Java Server Faces)

JavaのWebアプリケーションを作るためのフレームワークの一つ。ユーザーの文字入力や回答を選択するボックスなどのユーザーインターフェースを作ることに特化している。

1. ライブラリ

* Bootstrap

レスポンシブデザインに対応したフロントエンド(HTML/ CSS/ JS)のフレームワーク。フォームやボタンを作るのに非常に便利。

* Flyway

データベースのマイグレーションフレームワーク。マイグレーションとは異なる種類の装置やソフトウェア、システム、データ形式などの間でデータを移すこと（業務データの移行など）。データベースのスキーマ（テーブル）のバージョンを管理をする。

* Jackson

Javaで使用できるJSONライブラリのこと。JSONオブジェクトをJavaオフジェクトにマップしたり（関連付けること）その逆を行うことが出来る。言語を超えたデータ交換のため、データフォーマットとしてJSONを利用することが出来る。

* jQuery

JavaScriptのためのライブラリ。JavaScriptを簡単に記述できるようになり、WebサイトやWebアプリケーションで利用される。HTMLやCSSを簡単に操作することが出来る。

* Junit

Java開発で用いられるオープンソースのテストフレームワーク。これを用いることで、Junitが提供するテストメソッドを用いて、テストの期待値とテスト結果の比較、およびコード・カバレッジ（テスト対象となるプログラムコード全体の中でテスト済みの占める割合）の可視化が出来る。

* Log4j

オープンソースのJavaプログラム用ロギングユーティリティのこと。ロギングユーティリティとはプログラム内部からデバック情報や、エラー情報などのログデータ（）をコンソール、ファイル、その他ログサーバーなどへ出力する機能。大規模なシステムなどでもプログラム実行時の正確な流れを把握したりできる。

* Lombok

Java特有の凡長なコードを簡潔にしてくれるオープンソースのライブラリ。Getter, setter, toString,などの「何度も繰り返し書くコード」をコンパイル時に自動生成してくれる。

* ORM (Object-relational mapping)

データベース上のデータとプログラム上のデータとの相互交換を行うためのプログラミング技法のこと。データベースアクセスを簡易化し、プログラムのコードとして簡単にデータベース上のデータを扱うことが出来る。

1. 連絡ツール
2. ChatWork

ビジネスコミュニケーションに特化したチャットツール。チャット機能やファイル送信機能など、ビジネス利用をスムーズにするための機能が充実している。また、タスク管理機能など、自身の業務効率化にも繋がる。また、ビデオ通話にも対応しているため、会議もできる。

1. Slack

ビジネスコミュニケーションに特化したチャットツール。特徴としては、後から参加してもコミュニケーションの流れが追えること、送信した内容の編集、通知の設定を細かくできる、検索機能がとてもパワフル、外部サービスとの連携などがある。

1. IP Messenger

同じネットワーク内のメンバーを自動認識して、簡単にクライアント同士がリアルタイムに情報交換できるソフトウェア。外部サーバーへ接続せずメッセージやファイルのやり取りが出来、アカウントなどは必要ない。開発しているものがインターネットの制限がかかっている場合にこれがよく用いられる。

1. Java
2. Javadoc

ソースコードの内容から自動判定したり、特定の書式で書かれたコメントを抜き出すことによって自動生成された、ホームページっぽい形式になっているJavaのソースコードの説明書。ソースコードから説明書を作るときに使うのが、Javadocコマンド。説明書に載せたいコメントは/\*\*から始まる。

1. アノテーション

Annotationとは注釈という意味。Javadocは人に対しての説明書だが、アノテーションはコンパイラや実行環境に対してプログラムの内容を伝えるためのもの。これによって例えば、コンパイラで出力される警告メッセージを抑制したり、実行環境によって、プログラムの実行環境によってプログラムの動作を変更したりできる。大規模な開発を行うときは、「コーディングスタイルについてルールを決めて、そのルールに従ったコードを記述する」という方法がありますが、文書や口頭による申し合わせでは徹底できないことがあります。これを徹底させるためにアノテーションは利用できます。

1. メモリ管理

* ヒープ

データ構造の一種であり、木構造の完全2分木を用いたデータ構造である。最大ヒープと最小ヒープの２つ種類があり、根が最大値か最小値かという違いのみである。ヒープは優先度付きキュー（データの集合に含まれる各要素に優先度がついているキュー）として使用できる。キューとは列のこと。つまりデータの出入りの方法を意味している。また、コンピュータが利用するメモリ領域の種類の一つでもあり、実行時に任意のタイミングで確保や解放が可能なものもヒープという。（あるプログラムが使用できるメモリ領域のこと）

* GC

プログラムが動作するには2つの動作が必要なる。1つ目はプログラム処理に必要な情報をメモリ上に読み込むこと。2つ目は読み込んだ情報を元に処理を実行することである。処理を実行した後、読み込まれた情報は不要になるため、GC（Garbage Collection）は用いてこれを解放することでメモリ領域を管理している。

1. GoF

（Gang of Four）とはプログラミングとしてはオブジェクト指向プログラミングにおける再利用性の高いコーディングのパターン、デザインパターンをまとめた4人のプログラマのこと。この4人は、エーリヒ・ガンマ、リチャード・ヘルム、ラルフ・ジョンソン、ジョン・ブリシディースであり、Object-Oriented Softwareの著者である。その中で23種類のデザインパターンをまとめ、既定の規約を設けた。

1. デザインパターン

オブジェクト指向プログラミングで用いられる再利用性の高いコーディングのパターン。生成、構造、振る舞いの3つに分類されたパターンがGoFにある。

1. その他
2. オブジェクト指向

プログラムをモノとして捉える考え方。クラスを基にする。例えば、ゲームでキャラクターを作るときに1人1人に人間の構造を１つずつ用意するのは大変である。だが、人間の設計図を最初に作ってしまえば、そこに当てはめるだけになるので簡単になる。

1. バッチ処理

決められたタイミングで処理を一括して行うこと。主に利用されるものとして、日次処理など。指定された時間に自動で実行したり、ユーザの入力を必要とせずに動作する、終了時に実行結果が分かるようにするなど。

1. コンパイル

プログラミング言語で記述されたソフトウェアのソースコードをコンピューターが実行可能な形式（オブジェクトコードまたはバイナリコード）に変換する作業のこと。

1. カバレッジ

カバーしている範囲のこと。プログラムのテストの際によく使われる。テスト全体でこれくらいの確認をする必要がある中、現在どのくらいがテスト済みかを示す割合のこと。

1. プラットフォーム

土台となる環境のこと。ソフトウェアが動作するための土台として機能するもの（ソフトウェアにとってのＯＳやＯＳにとってのハードウェアなど）。

1. 業界用語