AxiZ
Academy eXtend IT skill Zero

個人開発の流れ

目的

個人開発の目的

1. 実際のシステム開発のイメージをつける
2. 期間が決められた開発に対するスケジュールの重要性を把握する
3. 報告／連絡／相談の重要性を把握する
4. プロジェクトにおける自身の役割と責任を理解する
5. 取組み姿勢／作業効率／プログラミングスキルなどの総合的な向上を図る
6. Gitを使ったプロジェクト管理
7. クラウド上へのデプロイ

目標

個人開発の目標

- 個人で決定したシステムを完成させる

  完成を最優先　現場の仕事をイメージすること。

  実際の仕事でもシステムが完成しなければ その顧客からは二度と仕事が来なくなる。

  また、たとえ完成しても不具合が多いシステムでは完成していないことと同じことである。

  丁寧にプログラミングを行い、十分なテストを行うこと。

目標

個人開発の目標

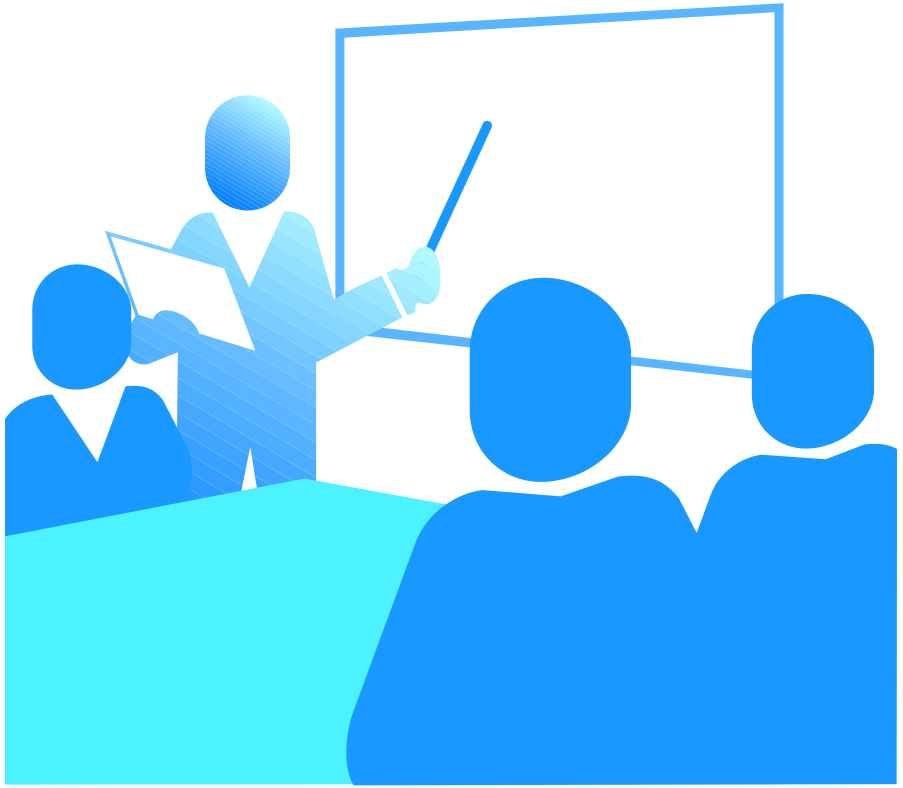
- 作業速度を理解してスケジュール通りに作業を進める
  - 作業スケジュールを十分理解し、作業スケジュールを組むこと
    最初に適当にスケジュールを組み、後で「無理でした」では意味がない。

個人作業

個人開発の流れ

個人開発は以下の流れで実施する。

1. 制作物決定
2. スケジュール作成
3. 要件定義
4. 基本設計書作成
5. 詳細設計書作成
6. プログラミング
7. テスト
8. デプロイ



個人作業

制作物決定

要件定義にて決定した制作物を作成する。

1. 内容重視
   プログラムは勿論だが、実際のシステムをイメージして作成することが目的となる。外観も意識し、CSSを適応すること。

2. プロジェクトとして意識
   個人開発を１つのプロジェクトとして意識すること。

個人作業

スケジュール作成

- 作業内容
  - スケジュールの作成
    - 作業者本人が納得したスケジュールを作成すること
    - スケジュールが完成したら講師に報告すること。

- レビュー対象
  - 開発スケジュール

個人作業

要件定義

1. 作業内容
   - 要件定義書の作成

2. レビュー対象
   - 要件定義書
   - その他、レビューに必要と感じる追加資料

※ 講師の承認が下りるまで次の作業に取り掛かることは認めない

個人作業

要件定義のポイント

1. お客様へ要件の説明

   どのような資料を用意すれば分かりやすいか、話を聞く側のことをよく考えて対応する。(時間は限られているため、講師が内容を理解できるレベルで調整)

2. レビューの進め方

   分かりづらい説明は再レビューの対象となる。

   レビュー時に講師から指摘された内容および対応が必要と思われる機能は、修正／追加を行うこと。

個人作業

基本設計書・テーブル定義書

1. 作業内容
   - 基本設計書作成
   - テーブル定義書作成

2. レビュー対象
   - 基本設計書テーブル定義書
   - その他、レビューに必要と感じる追加資料

※ 講師の承認が下りるまで次の作業に取り掛かることは認めない

個人作業

基本設計書・テーブル定義書のポイント

1. 顧客への説明
   基本設計書はお客様にも見てもらう資料です。
   お客様が理解できる内容にする。

2. データの洗い出し
   システムに必要なデータの洗い出しを行う。
   ここで管理すべきデータが漏れると後々、大きな手戻りになる。

個人作業

詳細設計書

1. 作業内容
   - 詳細設計書作成

2. レビュー対象
   - 詳細設計書
   - その他、レビューに必要と感じる追加資料

個人作業

- **詳細設計書のポイント**

1. 粒度
   - 詳細設計はプログラマーが確認してプログラミングを行うための資料です。
   - 細かい仕様を知らなくても、詳細設計書を見ることでプログラムが作れるレベルの粒度を目指す。

個人作業

- **プログラミング**

1. 作業内容
   - プログラミング

2. レビュー対象
   - ソースコード

個人作業

プログラミングのポイント

1. Springの使用
   強制はしませんが可能な限りSpringを使用してシステムを作成すること。

2. データベースを使用する
   テーブルではなく、データベースから作成すること。
   データの取り扱いには十分に注意すること。
   本番機を使用することで本番で失敗するリスクを減らすことができる。
   テストに使用したデータなどは、SQLで証拠を残しておく。

3. 処理の共通化
   同じ処理はなるべく共通化するように設計を行う。
   共通化を行うことでプログラミング、テストのコストに大きな影響がでる。

個人作業

- テスト

  単体テスト、結合テストをそれぞれ実施する。

1. テスト仕様書の作成
   - 「詳細設計書」から「単体テスト仕様書」を作成する
   - 「基本設計書」から「結合テスト仕様書」を作成する
   - テスト仕様書作成は各設計時に行うこと
   - テスト実施の際に項目が足りないと感じた場合、随時項目を追加すること

2. テスト実施
   - 上記で作成した仕様書をもとに行う。

個人作業

- テスト

3. 正常パターン／異常パターンの確認

   テストはシステム開発における非常に重要な工程である。

   いい加減にテストを実施せず、テストデータも正常パターン／異常パターンの結果が正確か、十分に確認すること。作成者以外がテストを行うことが望ましい。

例：

ログイン画面：	作成者Aさん	テスト実行者Bさん
メニュー画面：	作成者Cさん	テスト実行者Aさん

※ テスト仕様書のサンプルを参考にし、テスト項目を作成すること

※ エビデンスの取得／結果の提出を忘れないように

個人作業

- **ソース管理**

  ソースコードはGitを使用してバージョン管理を行う。
  おそらくEclipse上でGitを利用したほうが楽。

  - リモートリポジトリの使用
    - リモートリポジトリはGitHubを用いてソースコードを管理する
    - GitHubのアカウントは個人で作成する
    - リポジトリはパブリックにし、誰でも参照できるようにする

個人作業

- **デプロイ**

  テストが終了し、プログラムが完成したら、クラウド上にデプロイし、インターネット上からサービルが利用できる形にする。

  - クラウドサービスはHerokuを利用する
  - DB(PostgreSQL)の設定を行い、プログラムをデプロイし、動作検証までを実施する

個人作業

最終成果物

1. 納品物
   - スケジュール
   - 要件定義書
   - 基本設計書
   - テーブル定義書（環境構築用のSQL）
   - 詳細設計書
   - 単体テスト仕様書（実施結果 + エビデンス）
   - 結合テスト仕様書（実施結果 + エビデンス）
   - ソース（プロジェクトフォルダを提出）
   - GitHubとHerokuのURL

個人作業

- **最終成果物**

※ 印刷されることを意識してフォーマットを確認すること
※ 自分が資料を見る側の立場になってフォーマットやフォントを確認すること