Woodland Academy

# Unit5

- テストとは
- 仕様書作成

Shape Your Future

# 目次

# 開発の流れ

開発の計画やスコープ、使用する製品を決める

画面レイアウトや機能一覧などシステムの全体像を設計

基本設計の内容をよりPGMに近い形式で書き直す

プログラムの作成

要件定義 → 基本設計 → 詳細設計 → 製造 →

作成したプログラム単体の検証

作成したプログラムを一連の流れで検証

作成したシステムと周辺システムを連携させ検証

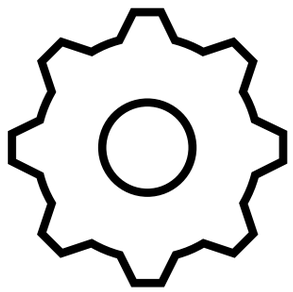
作成したシステムを本番環境へ移行

単体テスト → 結合テスト → システムテスト → 本番移行

Woodland Academy

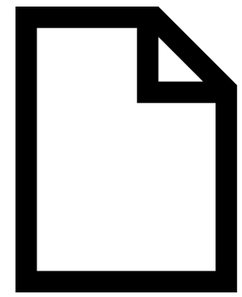
# テストとは

- 製造したプログラムが要件通り動作するか検証する段階



プログラム

要件通り動作するか検証



テスト結果

これは想定通り動作したからOK

これは不具合だから修正して再検証

次へ

前へ

単体テスト

検証

詳細設計

- プログラムが詳細設計書通り動作するか検証する
- 開発者が各々のプログラムを実行し、検証していく

結合テスト

検証

基本設計

- 画面は開発者が打鍵し、バッチはジョブ管理ツールを利用して実行することが多い
- 画面の入力～バッチの実行など、一連の流れを検証する

システムテスト

検証

要件定義

- テストケースは結合テストと似ているが、バッチ処理で周辺システムとの連携を行うようになる。
- PJによってはユーザーテストという形でエンドユーザーが検証に参加することもある

Woodland Academy

# その他のテスト

**ユーザーテスト**

・ユーザー側による受け入れテスト。主に操作性や数値の妥当性を検証する
・既存（代替）業務と比較することも多く、数値的なエラーも出てくる

**回帰テスト**

・一部機能を改修したときに、既存機能に影響がないか確認するテスト
・本命のテストより数が多くなることも…

**負荷テスト**

・画面やバッチ処理に負荷をかけ、性能面で問題が無いか検証するテスト
・同時接続やデータ量jの増加などの負荷をかけることが多い

**セキュリティテスト**

・SQLインジェクションなどの脆弱性が無いかを検証するテスト
・専門の業者に委託することが多い

Woodland Academy

# テストの流れ

テストする内容や条件を決める

テストケース設計

テスト実施に必要なデータの準備をする

データ準備

テストを実施、結果をエビデンスとして残す

検証

不具合がある場合修正して再テスト。

# テスト実施

画面 → 検証 → 実際に画面操作

- シナリオ通りに画面を操作し、画面の文言、挙動やDBの状態を検証する
- 画面の文言や挙動のエビデンスは画像を取得することが多い

バッチ → 検証 → バッチ処理実行

- データを準備した後、バッチ処理を実行してDBの状態やファイルの状態を確認する
- DBの状態の場合はExcelや画像で、ファイルの状態はZipで保存することもある。

帳票 → 検証 → 印刷して確認

- PDF生成や実際に紙に出力してレイアウトを確認する
- 少しのずれでもNGになることがあり、検証の難易度がかなり高い

Woodland Academy

# テストの大変なところ

- テストケースの洗い出し、およびそれに必要なデータの作成で単純に作業量（確認量）が多い
- 人間が行う作業なので、ミスもある（不具合が通過することがある）
- 単調な作業の時もある

# Q&A

# 目次

# テストケース設計の流れ

参照系ならこのようなテスト、更新系ならこのようなテスト、というPJとしての行うべき観点

テスト対象機能で必要な観点をピックアップ

対象機能に応じて境界値など具体的なケースを作成する

テスト観点作成 → テラーリング → 具体的なケース設計

# テスト観点

テスト観点作成 → テラーリング → 具体的なケース設計

テスト観点を整理する

どの機能で実施すべきかも整理しておくと使いやすい

過去のPJや会社の標準を流用すると良い

| テスト観点 | | | | | 画面 | | | | バッチ | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| テスト種別 | 大分類 | 中分類 | 小分類 | 目的 | 参照 | 更新 | ファイルDL | ファイルUL | 参照 | 追加 | 更新 | 削除 |
| 入力チェック | 正常系 | 検索 | 単項目検索 | 1つの検索条件のみで検索実行 | ○ | ー | ○ | ー | ー | ー | ー | ー |
| | | | 必須項目検索 | 必須の検索条件のみで検索実行 | ○ | ー | ○ | ー | ー | ー | ー | ー |
| | | | 全項目検索 | 全検索条件で検索実行 | ○ | ー | ○ | ー | ー | ー | ー | ー |
| | | 登録 | 任意項目未入力 | 最小限の入力項目で正常に登録できるか確認 | ー | ○ | ー | ○ | ー | ○ | ○ | ー |
| | | | 全項目入力 | 全項目入力した状態で登録できるか確認 | ー | ○ | ー | ○ | ー | ○ | ○ | ー |
| | | 文字列長 | 最大文字数入力 | 最大の桁数を入力した状態で登録できるか確認 | ー | ○ | ー | ○ | ー | ○ | ○ | ー |
| | 異常系 | 共通 | 必須チェック | 必須項目で未入力があるとエラーになることを確認 | ー | ○ | ー | ○ | ー | ○ | ○ | ー |
| | | 文字列 | 文字列長チェック | 最大桁数より多い桁数を入力できないことを確認 | ー | ○ | ー | ○ | ー | ○ | ○ | ー |
| | | | 整合性チェック | マスタに存在しないデータを指定されたときにエラー | ー | ○ | ー | ○ | ー | ○ | ○ | ー |
| | | | 郵便番号チェック | 不正な郵便番号が入力された場合にエラー | ー | ○ | ー | ○ | ー | ○ | ○ | ー |
| | | | 電話番号チェック | 不正な電話番号が入力された場合にエラー | ー | ○ | ー | ○ | ー | ○ | ○ | ー |
| | | 日付 | 日付形式チェック | 既定の形式以外で入力された場合エラー | ー | ○ | ー | ○ | ー | ○ | ○ | ー |
| | | | 日付範囲チェック | 2000年以前を指定された場合エラー | ー | ○ | ー | ○ | ー | ○ | ○ | ー |
| | | 数値 | 形式チェック | 数値以外が入力された場合エラー | ー | ○ | ー | ○ | ー | ○ | ○ | ー |
| | | | 小数点チェック | 小数点が〇〇以上指定された場合エラー | ー | ○ | ー | ○ | ー | ○ | ○ | ー |
| | | | 数値範囲チェック | 指定された範囲以外の数値が入力された場合エラー | ー | ○ | ー | ○ | ー | ○ | ○ | ー |
| 異常時の挙動 | 画面 | 画面上の挙動 | | エラー項目がハイライトされることを確認 | ー | ○ | ー | ○ | ー | ー | ー | ー |
| | | DB | | 更新されないことを確認する | ー | ○ | ー | ○ | ー | ー | ー | ー |
| | バッチ | 更新対象テーブル | | 処理開始前と変わらないことを確認 | ー | ー | ー | ー | ー | ○ | ー | ー |
| | | エラーログ | | バッチ処理エラーログが出力されることを確認 | ー | ー | ー | ー | ー | ○ | ○ | ○ |

# テラーリング

テスト観点作成 → テラーリング → 具体的なケース設計

個別機能でどの観点でテストをするか取捨選択する

| テスト観点 | | | | | テラーリング（画面） | | | テラーリング（バッチ） | | |
|---|---|---|---|---|---|---|---|---|---|---|
| テスト種別 | 大分類 | 中分類 | 小分類 | 目的 | A画面 | B画面 | C画面 | Aバッチ | Bバッチ | Cバッチ |
| 入力チェック | 正常系 | 検索 | 単項目検索 | 1つの検索条件のみで検索実行 | ー | ー | ー | ー | ー | ー |
| | | | 必須項目検索 | 必須の検索条件のみで検索実行 | ○ | ー | ー | ー | ー | ー |
| | | | 全項目検索 | 全検索条件で検索実行 | ー | ー | ○ | ー | ー | ー |
| | | 登録 | 任意項目未入力 | 最小限の入力項目で正常に登録できるか確認 | ー | ○ | ー | ○ | ○ | ー |
| | | | 全項目入力 | 全項目入力した状態で登録できるか確認 | ー | ○ | ー | ○ | ○ | ー |
| | | 文字列長 | 最大文字数入力 | 最大の桁数を入力した状態で登録できるか確認 | ー | ○ | ー | ○ | ○ | ー |
| | 異常系 | 共通 | 必須チェック | 必須項目で未入力があるとエラーになることを確認 | ー | ○ | ー | ○ | ○ | ー |
| | | 文字列 | 文字列長チェック | 最大桁数より多い桁数を入力できないことを確認 | ー | ○ | ー | ○ | ○ | ー |
| | | | 整合性チェック | マスタに存在しないデータを指定されたときにエラー | ー | ○ | ー | ○ | ○ | ー |
| | | | 郵便番号チェック | 不正な郵便番号が入力された場合にエラー | ー | ○ | ー | ○ | ○ | ー |
| | | | 電話番号チェック | 不正な電話番号が入力された場合にエラー | ー | ○ | ー | ○ | ○ | ー |
| | | 日付 | 日付形式チェック | 既定の形式以外で入力された場合エラー | ー | ○ | ー | ー | ○ | ー |
| | | | 日付範囲チェック | 2000年以前を指定された場合エラー | ー | ○ | ー | ー | ー | ー |
| | | 数値 | 形式チェック | 数値以外が入力された場合エラー | ー | ○ | ー | ー | ー | ー |
| | | | 小数点チェック | 小数点が○○以上指定された場合エラー | ー | ○ | ー | ー | ー | ー |
| | | | 数値範囲チェック | 指定された範囲以外の数値が入力された場合エラー | ー | ○ | ー | ー | ー | ー |
| 異常時の挙動 | 画面 | 画面上の挙動 | | エラー項目がハイライトされることを確認 | ー | ○ | ー | ー | ー | ー |
| | | DB | | 更新されないことを確認する | ー | ○ | ー | ー | ー | ー |
| | バッチ | 更新対象テーブル | | 処理開始前と変わらないことを確認 | ー | ー | ー | ○ | ○ | ー |
| | | エラーログ | | バッチ処理エラーログが出力されることを確認 | ー | ー | ー | ○ | ○ | ○ |

例えば日付が無いなど個別機能に合わせる

# テストケース設計

テスト観点作成 → テラーリング → 具体的なケース設計

## テスト方針

**ブラックボックス**

- 内部構造は考慮せず、業務要件に沿っているかの検証を行うテスト
- 結合テストやシステムテストで採用することが多い

**ホワイトボックス**

- PGMのすべての経路を通ように行う網羅的なテスト
- 単体テストで採用することが多い

## 設計技法

◆テストケースの考え方
- 観点整理：事前に用意したテスト観点を参考に設計
- 同値分割：同じ結果になる値群のうち、１つのみ検証
- 境界値：条件の閾値の前後の値で検証
- デジョンテーブル：仕様を整理する表

◆テストケースの考え方
- 直交表：直交表を用いて２つの組み合わせを全網羅
- ALL-Pair法：ツールを用いて２つの組み合わせ全網羅

※直交表は各要素が同数出現、ALL-Pairは１回以上

※ ALL-Pairの方がケース数は少なく直交表の方が網羅性は高い

Woodland Academy

# テストケース設計

テスト観点作成 → テラーリング → 具体的なケース設計

## 仕様

| 入力値 | 挙動 |
|---|---|
| 負の値 | エラー |
| 少数 | エラー |
| 0 | A |
| 1～９９ | B |
| １００以上 | C |

テストケース例
- 負の値
- 小数
- 0
- 1
- 99
- 100

## 同値分割・境界値

| ルール | | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
|---|---|---|---|---|---|---|---|---|---|
| 条件 | 条件1 | Y | Y | Y | N | Y | N | N | N |
| | 条件2 | Y | Y | N | Y | N | N | Y | N |
| | 条件3 | Y | N | Y | Y | N | Y | N | N |
| 結果 | 結果1 | Y | Y | Y | - | - | - | Y | - |
| | 結果2 | - | - | | Y | - | Y | - | - |
| | 結果3 | - | - | - | - | - | - | - | Y |

7つの要素があり、
各要素には2つの選択肢がある

| 試行回数 | 要因 | | | | | | |
|---|---|---|---|---|---|---|---|
| | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
| 実験1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| 実験2 | 1 | 1 | 1 | 2 | 2 | 2 | 2 |
| 実験3 | 1 | 2 | 2 | 1 | 1 | 2 | 2 |
| 実験4 | 1 | 2 | 2 | 2 | 2 | 1 | 1 |
| 実験5 | 2 | 1 | 2 | 1 | 2 | 1 | 2 |
| 実験6 | 2 | 1 | 2 | 2 | 1 | 2 | 1 |
| 実験7 | 2 | 2 | 1 | 1 | 2 | 2 | 1 |
| 実験8 | 2 | 2 | 1 | 2 | 1 | 1 | 2 |

## L8直交表

Woodland Academy

# テストケースを考えてみよう

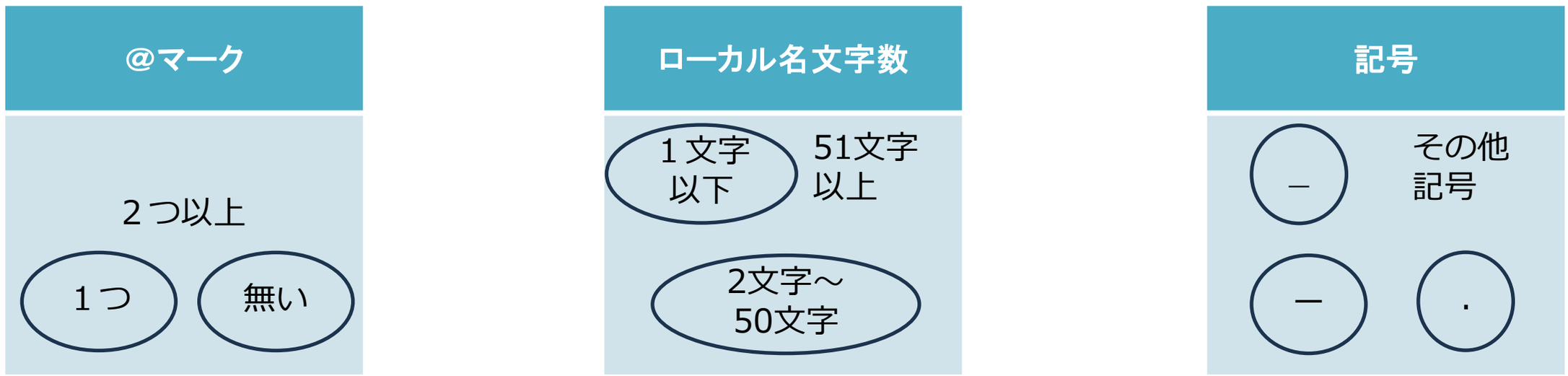
- 以下の画面のテストケースを考える


- メールアドレスは「ローカルパート@ドメインパート」で構成される。
ローカルパートは2文字以上50文字以下の半角英数字と記号。
使用可能な記号は-_.のみとする。
ドメインパートはローカルパートと同様の文字で有無のみ確認する。
メールアドレスが仕様に合致し、2つの入力文字列が一致した場合は正常とする。

# 同値クラスを作る

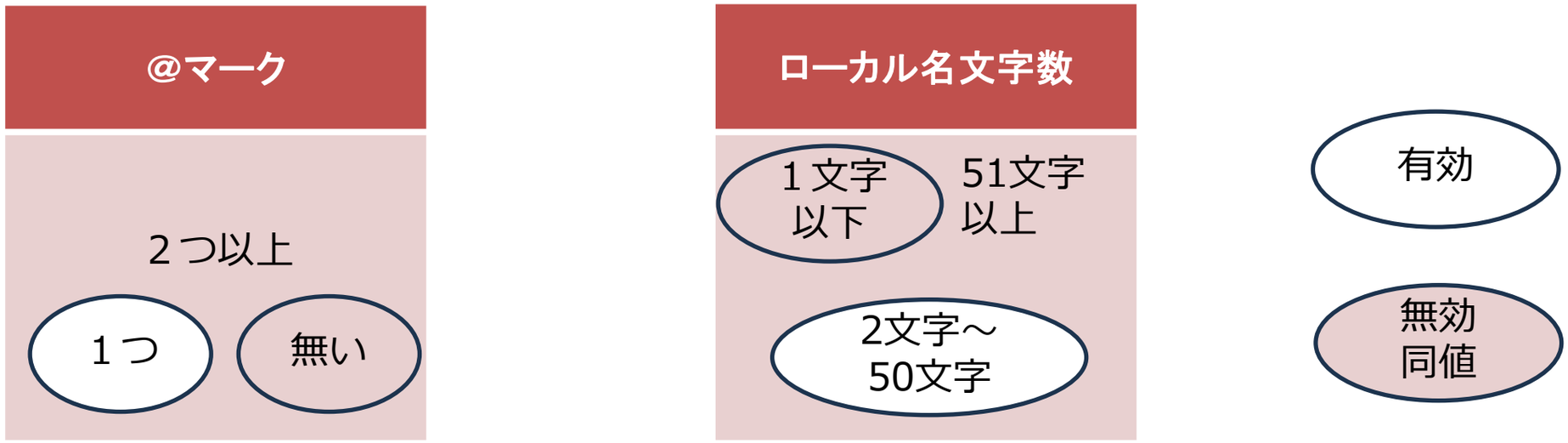
- テスト結果をグループ分けすることをここでは「同値クラスを作る」といいます。通常、グループ分けの基準は一通りではなく複数あります。その基準をここでは「観点」といいます。

メールアドレスについては、例えば「アットマークの数」「文字種別」「文字数」「記号の有無」「メールアドレスの一致性」などの観点を挙げることができます。



# 同値クラスを分類する

- 同値クラスには、それらが影響を及ぼすテスト結果によって「有効同値クラス」「無効同値クラス」に分類できます。
- 有効同値クラスは、そのテスト結果を正常系にするであろう同値クラスをいいます。
- 一方で、無効同値クラスは、そのテスト結果を準正常系・異常系にするであろう同値クラスをいいます。
- なお、正常系・準正常系・異常系は以下のような使い方をしています。



# 代表値を決定する

- それぞれの観点に対して、有効同値クラスと無効同値クラスが何であるかを考えた後は、実際にテストする時をイメージして、テスト条件を決定します。具体的には数値であったり、文字列であったり、選択肢であったりします。

| | 同値クラス1 | 同値クラス2 | 同値クラス3 | 同値クラス4 |
|---|---|---|---|---|
| アットマーク | 1個 | 2個以上 | 0個 | |
| | 1個 | **2個** | **0個** | |
| ローカルパート文字数 | 2文字以上50文字以下 | 1文字以下 | 51文字以上 | |
| | 10文字 | **1文字以下** | **60文字** | |
| 記号 | アンダースコア | ハイフン | ピリオド | パーセント |
| | 「_」を含む | 「-」を含む | 「.」を含む | **「%」を含む** |
| 文字種別 | a~z | A~Z | 0~9 | その他の文字 |
| | 「test」を含む | 「SOFT」を含む | 「123」を含む | **「テスト」を含む** |
| ドメインパート | あり | なし | | |
| | 「softest.jp」 | **「なし」** | | |

**赤字**は無効同値クラスの代表値

Woodland Academy

# テスト条件を決定する

- 観点が一つの場合は、決定した代表値をそのままテスト条件とします。観点が複数ある場合は、**無効同値クラスが重ならないように**テスト条件を組み合わせてテスト条件とします。

| テスト番号 | アットマーク | ローカルパート文字数 | 記号 | 文字種別 | ドメインパート | 期待結果 |
|---|---|---|---|---|---|---|
| 1 | 1個 | 10文字 | 「_」を含む | 「test」を含む | 「softest.jp」 | 正常処理 |
| 2 | **2個** | 10文字 | 「_」を含む | 「test」を含む | 「softest.jp」 | **エラー処理** |
| 3 | **0個** | 10文字 | 「_」を含む | 「test」を含む | 「softest.jp」 | **エラー処理** |
| 4 | 1個 | **1文字** | 「_」を含む | 「test」を含む | 「softest.jp」 | **エラー処理** |
| 5 | 1個 | **60文字** | 「_」を含む | 「test」を含む | 「softest.jp」 | **エラー処理** |
| 6 | 1個 | 10文字 | 「-」を含む | 「test」を含む | 「softest.jp」 | 正常処理 |
| 7 | 1個 | 10文字 | 「.」を含む | 「test」を含む | 「softest.jp」 | 正常処理 |
| 8 | 1個 | 10文字 | **「%」を含む** | 「test」を含む | 「softest.jp」 | **エラー処理** |
| 9 | 1個 | 10文字 | 「_」を含む | 「SOFT」を含む | 「softest.jp」 | 正常処理 |
| 10 | 1個 | 10文字 | 「_」を含む | 「123」を含む | 「softest.jp」 | 正常処理 |
| 11 | 1個 | 10文字 | 「_」を含む | **「テスト」を含む** | 「softest.jp」 | **エラー処理** |
| 12 | 1個 | 10文字 | 「_」を含む | 「test」を含む | **なし** | **エラー処理** |

**赤字**は無効同値クラスの代表値

次へ

- テスト番号1はすべての観点で有効同値クラスの代表値を選択しています。テスト番号2～3はアットマークという観点で無効同値クラスの代表値を選択し、テスト番号4～5はローカルパート文字列という観点で無効同値クラスの代表値を選択しています。以下、同じようにそれぞれの観点で代表値を選択しながら、テスト条件を並べていきます。

| テスト番号 | アットマーク | ローカルパート文字数 | 記号 | 文字種別 | ドメインパート | 期待結果 |
|---|---|---|---|---|---|---|
| 1 | 1個 | 10文字 | 「_」を含む | 「test」を含む | 「softest.jp」 | 正常処理 |
| 2 | **2個** | 10文字 | 「_」を含む | 「test」を含む | 「softest.jp」 | **エラー処理** |
| 3 | **0個** | 10文字 | 「_」を含む | 「test」を含む | 「softest.jp」 | **エラー処理** |
| 4 | 1個 | **1文字** | 「_」を含む | 「test」を含む | 「softest.jp」 | **エラー処理** |
| 5 | 1個 | **60文字** | 「_」を含む | 「test」を含む | 「softest.jp」 | **エラー処理** |
| 6 | 1個 | 10文字 | 「-」を含む | 「test」を含む | 「softest.jp」 | 正常処理 |
| 7 | 1個 | 10文字 | 「.」を含む | 「test」を含む | 「softest.jp」 | 正常処理 |
| 8 | 1個 | 10文字 | **「%」を含む** | 「test」を含む | 「softest.jp」 | **エラー処理** |
| 9 | 1個 | 10文字 | 「_」を含む | 「SOFT」を含む | 「softest.jp」 | 正常処理 |
| 10 | 1個 | 10文字 | 「_」を含む | 「123」を含む | 「softest.jp」 | 正常処理 |
| 11 | 1個 | 10文字 | 「_」を含む | **「テスト」を含む** | 「softest.jp」 | **エラー処理** |
| 12 | 1個 | 10文字 | 「_」を含む | 「test」を含む | **なし** | **エラー処理** |

**赤字**は無効同値クラスの代表値

# テスト条件を見直す

- テスト実施するための情報はほとんど揃っています。最後に、これらの情報を整理したり、修正したり、追加したりします。ここではこの作業を「見直し」といいます。

| テスト番号 | アットマーク | ローカルパート文字数 | 記号 | 文字種別 | ドメインパート | 期待結果 |
|---|---|---|---|---|---|---|
| 1 | 1個 | 10文字 | 「_」を含む | 「test」を含む | 「softest.jp」 | 正常処理 |
| 2 | **2個** | 10文字 | 「_」を含む | 「test」を含む | 「softest.jp」 | **エラー処理** |
| 3 | **0個** | 10文字 | 「_」を含む | 「test」を含む | 「softest.jp」 | **エラー処理** |
| 4 | 1個 | **1文字** | 「_」を含む | 「test」を含む | 「softest.jp」 | **エラー処理** |
| 5 | 1個 | **60文字** | 「_」を含む | 「test」を含む | 「softest.jp」 | **エラー処理** |

- 例えば、ローカルパート文字列が1文字であって、文字列として「test」を含むことは不可能です。そこで、このテスト条件では何を確認したいのかを思い出してみます。確認したいのは「**ローカルパートが1文字ではいけないので、エラー処理となるかどうか**」です。従って、ローカルパートが1文字であることは維持しながら、文字を「」であったり「t」といった条件に修正します。あるいは、文字列を「」の場合と「t」の場合の2回分に追加するのもよいです。

Woodland Academy

# 修正後のテスト条件

| テスト番号 | アットマーク | ローカルパート文字数 | 記号 | 文字種別 | ドメインパート | メールアドレス文字列 | 期待結果 |
|---|---|---|---|---|---|---|---|
| 1 | 1個 | 10文字 | 「_」を含む | 「test」を含む | 「softest.jp」 | test_98765@softest.jp | 正常処理 |
| 2 | **2個** | 10文字 | 「_」を含む | 「test」を含む | 「softest.jp」 | test_98765@@softest.jp | **エラー処理** |
| 3 | **0個** | 10文字 | 「_」を含む | 「test」を含む | 「softest.jp」 | test_98765softest.jp | **エラー処理** |
| 4 | 1個 | **1文字** | 「_」だけ | | 「softest.jp」 | _@softest.jp | **エラー処理** |
| 5 | 1個 | **1文字** | | 「t」だけ | 「softest.jp」 | t@softest.jp | **エラー処理** |
| 6 | 1個 | **60文字** | 「_」を含む | 「test」を含む | 「softest.jp」 | test_98765~~@softest.jp | **エラー処理** |
| 7 | 1個 | **なし** | | | 「softest.jp」 | @softest.jp | **エラー処理** |
| 8 | 1個 | 10文字 | 「ー」を含む | 「test」を含む | 「softest.jp」 | test-98765@softest.jp | 正常処理 |
| 9 | 1個 | 10文字 | 「.」を含む | 「test」を含む | 「softest.jp」 | test.98765@softest.jp | 正常処理 |
| 10 | 1個 | 10文字 | **「%」を含む** | 「test」を含む | 「softest.jp」 | test%98765@softest.jp | **エラー処理** |
| 11 | 1個 | 10文字 | 「_」を含む | 「SOFT」を含む | 「softest.jp」 | SOFT_98765@softest.jp | 正常処理 |
| 12 | 1個 | 10文字 | 「_」を含む | 「123」を含む | 「softest.jp」 | 123_987654@softest.jp | 正常処理 |
| 13 | 1個 | 10文字 | 「_」を含む | **「テスト」を含む** | 「softest.jp」 | テスト_987654@softest.jp | **エラー処理** |
| 14 | 1個 | 10文字 | 「_」を含む | 「test」を含む | **なし** | test_1234@ | **エラー処理** |

**赤字**は無効同値クラスの代表値

# 境界値分析とは

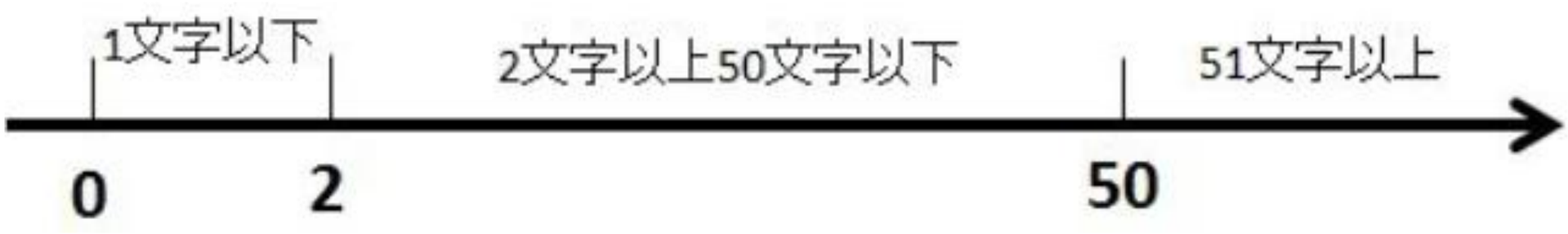
- 同値分割で得られた同値クラスの境界や端、その近くに注目してテスト条件を考えるブラックボックステスト技法の一つです。テスト設計で同値分割を活用する大きな理由は、効率的にバグを見つけるためですが、境界値分析を活用する大きな理由は、効果的にバグを見つけるためです。
- 一般的には「境界値にバグが多く存在する」と言われていますが、境界値のそばにもバグは多く存在し、未知の境界や余分な境界が引き起こすバグも少なくありません。ここでは、これら3つの注目ポイントに沿って解説します。

# 境界値のバグ

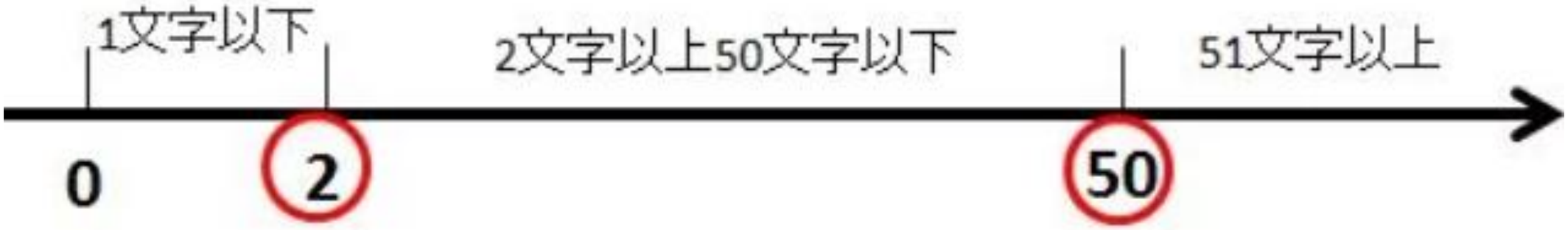
- ローカルパートの文字数という観点で「2文字以上50文字以下の半角英数字と記号」という仕様から、同値クラスを4つ作りました。

| ローカルパート文字数 | 2文字以上50文字以下 | 1文字以下 | 51文字以上 | なし |
|---|---|---|---|---|
| | 10文字 | **1文字以下** | **60文字** | **なし** |

- 文字数のような連続性を持つ同値クラスの場合は、「数直線」のようなモデルを描くとわかりやすいです。2文字は有効同値クラスに最小値として属し、50文字も有効同値クラスに最大値として属します。

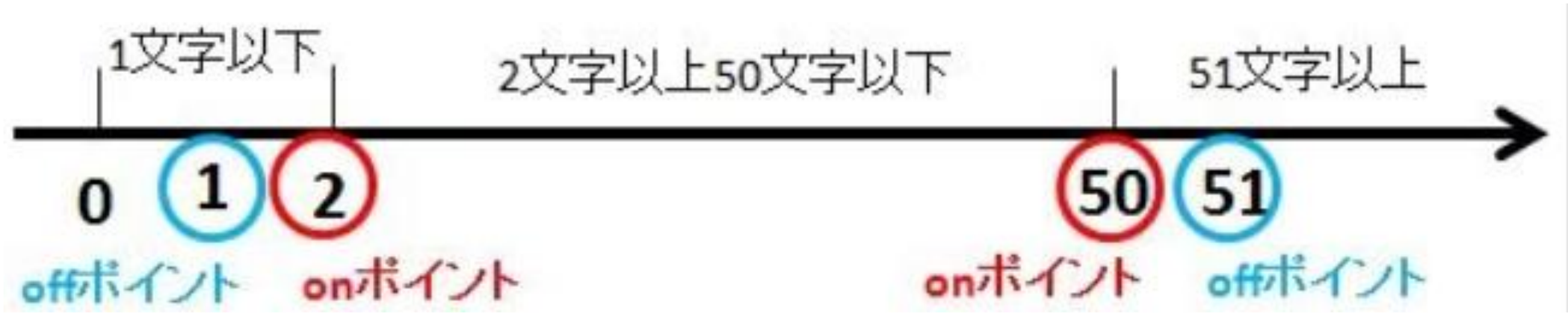


- 有効同値クラスの最大値および最小値が存在する場合、それらを「**onポイント**」と呼び、テストすべきです。数学的には最大値（あるいは最小値）が存在しない場合もありますが、その場合は上界（あるいは下界）の最小値（あるいは最大値）をonポイントと考えて、テストするとよいでしょう。



# 境界値近くのバグ

- 境界地近くのバグについても、同様にローカルパート文字数という観点で解説します。有効同値クラス（2文字以上50文字以下）と無効同値クラス（51文字以上）はonポイント（50文字）で接しています。無効同値クラスの中でonポイントに最も近い値を「**offポイント**」と呼び、テストすべきです。



# 今までの手順で作成したテスト条件

| テスト番号 | アットマーク | ローカルパート文字数 | 記号 | 文字種別 | ドメインパート | メールアドレス文字列 | 期待結果 |
|---|---|---|---|---|---|---|---|
| 1 | 1個 | 10文字 | 「_」を含む | 「test」を含む | 「softest.jp」 | test_98765@softest.jp | 正常処理 |
| 2 | **2個** | 10文字 | 「_」を含む | 「test」を含む | 「softest.jp」 | test_98765@@softest.jp | **エラー処理** |
| 3 | **0個** | 10文字 | 「_」を含む | 「test」を含む | 「softest.jp」 | test_98765softest.jp | **エラー処理** |
| 4 | 1個 | **1文字** | 「_」だけ | | 「softest.jp」 | _@softest.jp | **エラー処理** |
| 5 | 1個 | **1文字** | | 「t」だけ | 「softest.jp」 | t@softest.jp | **エラー処理** |
| 6 | 1個 | **60文字** | 「_」を含む | 「test」を含む | 「softest.jp」 | test_98765432109876543210987654321098765432109876543210987~0987654321~@softest.jp | **エラー処理** |
| 7 | 1個 | **なし** | | | 「softest.jp」 | @softest.jp | **エラー処理** |
| 8 | 1個 | 10文字 | 「ー」を含む | 「test」を含む | 「softest.jp」 | test-98765@softest.jp | 正常処理 |
| 9 | 1個 | 10文字 | 「.」を含む | 「test」を含む | 「softest.jp」 | test.98765@softest.jp | 正常処理 |
| 10 | 1個 | 10文字 | **「%」を含む** | 「test」を含む | 「softest.jp」 | test%98765@softest.jp | **エラー処理** |
| 11 | 1個 | 10文字 | 「_」を含む | 「SOFT」を含む | 「softest.jp」 | SOFT_98765@softest.jp | 正常処理 |
| 12 | 1個 | 10文字 | 「_」を含む | 「123」を含む | 「softest.jp」 | 123_987654@softest.jp | 正常処理 |
| 13 | 1個 | 10文字 | 「_」を含む | **「テスト」を含む** | 「softest.jp」 | テスト_987654@softest.jp | **エラー処理** |
| 14 | 1個 | 10文字 | 「_」を含む | 「test」を含む | **なし** | test_1234@ | **エラー処理** |
| 15 | 1個 | 2文字 | 「_」を含む | | 「softest.jp」 | _1@softest.jp | 正常処理 |
| 16 | 1個 | 50文字 | 「_」を含む | 「test」を含む | 「softest.jp」 | test_98765432109876543210987654321098765432109876543210987~5@softest.jp | 正常処理 |
| 17 | 1個 | 51文字 | 「_」を含む | 「test」を含む | 「softest.jp」 | test_98765432109876543210987654321098765432109876543210987~~5@softest.jp | **エラー処理** |

# テスト仕様書見本

- 以下の見本を見ながら、個人開発で作成した作品のユーザ登録とユーザのログインの単体テストと結合テストの仕様書を書いてみよう！

単体テスト仕様書見本

https://x.gd/qtxnT

結合テスト仕様書見本

https://x.gd/Zb4Cb

- 単体テストテストが書き終わったら講師に確認してもらい、ＯＫがでたら結合テストに入り、結合テストも書き終わったら、また講師に確認してもらおう。

Woodland Academy

Woodland Academy

# Q&A

Woodland Academy

# Thank you!

From Seeds to Woodland — Shape Your Future.

Shape Your Future