

# タスク管理アプリ

## ～Java 版～

作成日

2024/12/15

作成者

都野浩実

### プロジェクト概要

#### 1. 目的

- 自分専用のタスク管理システムを開発し、タスクの作成、編集、削除、および閲覧を効率的に行えるようにする。

#### 2. 背景

- 以前に Python で作成したタスク管理アプリケーションを基に、さらに機能を追加し、カスタマイズ性を高めた Java バージョンを作成することにした。

#### 3. 目標

- タスクの効率的な管理
- 使いやすいインターフェース
- 信頼性とパフォーマンス
- レスポンスデザイン

### ターゲットユーザー

#### 1. ユーザー層

- 一般のユーザー

#### 2. ユーザーのニーズ

- カスタマイズ性: テーマと配色の変更・インターフェースの柔軟性
- 機能性: タスクの効率的な管理・カテゴリ管理・検索機能
- 視覚的な魅力: 美しいデザイン・テーマの切り替え
- データの保護: 作成されたタスクのデータが安全に保存される

## 機能要件

### 1 主な機能



#### タスク管理

- ＊ タスクの作成: 新しいタスクを作成、タイトル・説明・優先度・カテゴリ・締切日を設定
- ＊ タスクの編集: 既存のタスクのタイトル・説明・優先度・カテゴリ・締切日を編集
- ＊ タスクの削除: 不要なタスクを削除し、削除済みタスクのリストに移動する。
- ＊ タスクの復元: 削除済みタスクを復元し、再度タスクとして表示する。
- ＊ タスクの完了/未完了の切り替え: タスクを完了状態、未完了状態に変更できる。



#### カテゴリ管理

- ＊ カテゴリの作成: タスクを分類するためのカテゴリを新たに追加できる。
- ＊ カテゴリの表示: 作成済みのカテゴリを表示し、タスク作成時に選択できる。



#### 検索機能

- ＊ タスクの検索: キーワードを入力して、タイトル、説明、カテゴリに基づいてタスクを検索できる。
- ＊ 削除済みタスクの検索: キーワードを入力して、削除済みタスクを検索できる。



#### インターフェースのカスタマイズ

- ＊ テーマの切り替え: ライトテーマとダークテーマの間で表示モードを切り替えられる。
- ＊ 差し色の変更: ユーザーが好みに応じて差し色を変更できる。
- ＊ デフォルト設定へのリセット: カスタマイズしたテーマと差し色をデフォルトに戻す機能。

### 2 非機能要件



#### レスポンスデザイン

- ＊ デバイス対応: スマートフォンやタブレットなど、さまざまなデバイスで快適に使用できるデザインを提供。



#### ユーザーインターフェースの使いやすさ

- ＊ 直感的な操作: 誰でも簡単に操作できるシンプルで直感的なインターフェースを提供。
- ＊ 視覚的な魅力: 見やすく、使いやすいデザインを提供。



#### システムパフォーマンス

- ＊ 高速な応答性: 高速な応答時間を確保し、ユーザーがストレスなく操作できるようにする。
- ＊ 信頼性: システムの安定性を確保し、データの消失やシステムエラーを最小限に抑える。



#### データの保護

- ＊ セキュリティ機能: タスクデータのセキュリティを確保し、個人情報を保護する仕組みを提供。
- ＊ データバックアップ: 定期的なデータバックアップを行い、データの消失に備える。

## システム構成

### 1 アーキテクチャ


🚦 フロントエンドとバックエンドを統合したアーキテクチャを採用

### 2 技術スタック

🚦 プログラミング言語: Java 21 

🚦 バックエンドフレームワーク: サーブレット技術

🚦 データベース: PostgreSQL 42.7.4 

🚦 アプリケーションサーバー: Tomcat 10 

🚦 フロントエンド技術:

✱ HTML5




✱ CSS3



✱ JavaScript



🚦 開発環境: Eclipse 2024-09 (4.33.0) 

## 開発スケジュール

日付	フェーズ	目標
12/7	設計	要件定義とシステムの設計の完了、基本的アーキテクチャの決定
12/8	設計	UI のデザインモックアップ作成、画面のレイアウト決定
12/9	設計	データベーススキーマを設計、テーブルとリレーションシップ定義
12/10	開発	フロントエンドの構築を開始し、HTML と CSS で主要なページを作成
12/11	開発	バックエンドの構築開始、サーブレットとデータベース接続を実装
12/12	開発	タスクの作成・編集・削除機能実装、UI のカスタマイズの完了
12/13	開発	テーマ・差し色の変更機能実装、UI のカスタマイズ完了
	テスト	ユニット・結合テストの実施、各機能の正確な動作の確認
	デバック	テストで発見されたバグを修正、システム全体の安定性の確認

## リスクと対策

### 1 インターフェースの互換性問題

🚦 リスク: 異なるデバイスやブラウザで表示や動作に問題が発生する可能性。

🚦 対策: 複数のデバイスやブラウザでテストを行い、互換性を確認する。  
レスポンシブデザインの原則に基づいて設計を行う。

---

## 品質と管理

---

### 1 テスト戦略

- ✚ 単体テスト:各機能やメソッドが単独で正しく動作することを確認する。  
(タスクの作成、編集、削除などの基本機能)
- ✚ 統合テスト:複数のコンポーネントやモジュールが正しく連携して動作することを確認する。  
(サーブレットとデータベースの連携、タスクの検索機能、カテゴリの管理機能)
- ✚ UI テスト:UI が直感的で操作しやすいことを確認する  
(テーマの切り替えや差し色の変更機能、レスポンシブデザイン)
- ✚ 機能テスト:全体的な機能が仕様通りに動作することを確認

### 2 品質基準

- ✚ バグなし:すべての機能が仕様通りに動作、バグが発生しないこと
- ✚ 直感的で操作しやすい UI:  
ユーザーが直感的に操作できる使いやすいインターフェース  
(  
テーマ切り替えや差し色変更のカスタマイズ機能において、簡単に操作できること)
- ✚ 高いパフォーマンス:  
アプリケーションが高速に応答し、スムーズに動作すること  
(タスクの検索やフィルタリング機能)
- ✚ 信頼性:システムが安定して動作し、データが確実に保存・管理されること

---

## 納品物

---

### 1 納品物の概要

- ✚ タスク管理アプリの完全版

### 2 納品予定日

- ✚ 2024/12/18

---

## レビューと反省

---

大まかスケジュール通りに進められたが、いろいろな機能を追加したことで Python 版と Java ではここまでコードに差が出るのかと痛感しました。

# 詳細設計書

## 機能構成

	機能一覧	機能の説明
1	タスク管理	タスクの作成・編集・削除 タスクの復元 タスクの完了/未完了の切り替え
2	カテゴリ管理	カテゴリの作成 カテゴリの表示
3	検索機能	タスクの検索 削除済みタスクの検索
4	UIのカスタマイズ	テーマの切り替え 差し色の変更

## シーケンス

- 🚩 ユーザー -> ブラウザ: タスク作成リクエスト送信
- 🚩 ブラウザ -> サーバー: HTTP リクエスト送信
- 🚩 サーバー -> TaskServlet: リクエスト処理
- 🚩 TaskServlet -> TaskDAO: タスク挿入要求
- 🚩 TaskDAO -> PostgreSQL データベース: 新しいタスクを挿入
- 🚩 PostgreSQL データベース -> TaskDAO: 挿入結果返却
- 🚩 TaskDAO -> TaskServlet: 挿入結果返却
- 🚩 TaskServlet -> ブラウザ: HTTP レスポンス返却
- 🚩 ブラウザ -> ユーザー: タスク作成成功通知

テーブル定義

1. タスクテーブル(tasks)

カラム名	データ型	制約	説明
<i>id</i>	INTEGER	PRIMARY KEY, AUTO_INCREMENT	ID
<i>title</i>	VARCHAR(255)	NOT NULL	タイトル
<i>description</i>	TEXT		説明
<i>completed</i>	BOOLEAN	DEFAULT FALSE	完了状態
<i>priority</i>	INTEGER	DEFAULT 0	優先度
<i>category</i>	VARCHAR(255)		カテゴリ
<i>deadline</i>	TIMESTAMP		締切日

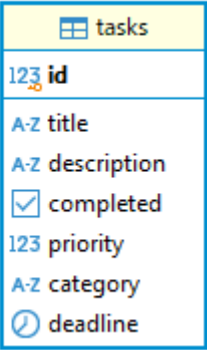
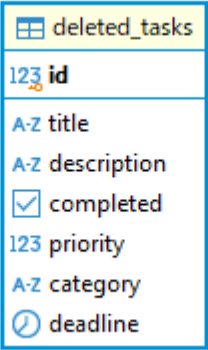
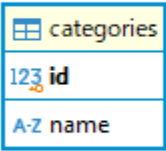
2. 削除済タスクテーブル (deleted\_tasks)

カラム名	データ型	制約	説明
<i>id</i>	INTEGER	PRIMARY KEY, AUTO_INCREMENT	ID
<i>title</i>	VARCHAR(255)	NOT NULL	タイトル
<i>description</i>	TEXT		説明
<i>completed</i>	BOOLEAN	DEFAULT FALSE	完了状態
<i>priority</i>	INTEGER	DEFAULT 0	優先度
<i>category</i>	VARCHAR(255)		カテゴリ
<i>deadline</i>	TIMESTAMP		締切日

3. カテゴリテーブル (categories)

カラム名	データ型	制約	説明
<i>id</i>	INTEGER	PRIMARY KEY, AUTO_INCREMENT	カテゴリ ID
<i>Name</i>	VARCHAR(225)	NOT NULL, UNIQUE	カテゴリ名

ER 図



## テスト表

テストケース番号	テスト項目	テスト内容	テストデータ	期待結果	実行結果
TC-01	カテゴリの追加	新しいカテゴリを作成し、「保存」ボタンをクリック	カテゴリ: "個人"	新しいカテゴリが生成され、カテゴリのプルダウンに表示される	○
TC-02	タスク作成	タスクフォームで新しいタスクを作成し、「保存」ボタンをクリック	タイトル: "買い物" 詳細: "スーパーで買い物" 優先度: 中 カテゴリ: "個人" 締切日: "2024-12-20"	新しいタスクが作成され、タスク一覧・詳細のポップアップに表示される	○
TC-03	タスクフォームのクリア	タスクフォームのクリアボタンを押すと入力中の項目がすべてクリアされる	タイトル: "買い物" 詳細: "スーパーで買い物" を入力した後にクリアする	入力フォームがデフォルトの状態に戻る	○
TC-04	タスク編集	既存のタスクを編集し、「保存」ボタンをクリック	新しいタイトル: "買い物リスト", 新しい詳細: "スーパーで必要なもの"	編集内容が保存され、タスク一覧・詳細のポップアップに反映される	○
TC-05	タスク削除	タスク一覧でタスクを選択し、「削除」ボタンをクリック	削除するタイトル: 買い物リスト	タスクが削除され、削除済みタスク一覧に移動される	○
TC-06	削除済みタスクへ画面遷移	「削除済みタスク」のボタンをクリック		削除済みタスクの画面へ遷移する	○
TC-07	タスク復元	削除済みタスク一覧でタスクを選択し、「復元」ボタンをクリック	復元するタイトル: 買い物リスト	タスクが復元され、再びタスク一覧に表示される	○
TC-08	タスク一覧に戻る	「タスク一覧に戻る」ボタンをクリック		タスクリスト画面に遷移する	○
TC-09	タスク完了/未完了切り替え	タスク一覧で「完了/未完了」ボタンをクリック	切り替えるタイトル: 買い物リスト	タスクの完了状態が正しく切り替わり、タスク一覧に反映される	○

テストケース番号	テスト項目	テスト内容	テストデータ	期待結果	実行結果
TC-10	タスクの検索	タスク検索画面でキーワードを入力し、「検索」ボタンをクリック	キーワード: “買い物”	検索条件に一致するタスクが一覧に表示される	○
TC-11	テーマ切り替え	設定画面でテーマを選択し、ライトテーマまたはダークテーマに切り替える	テーマ: “ダークテーマ”	選択したテーマがアプリケーション全体に適用される	○
TC-12	差し色変更	設定画面で差し色を選択し、好きな差し色に変更する	差し色: “レッド”	選択した差し色がアプリケーションの UI に反映される	○
TC-13	デフォルトに戻す	「デフォルトに戻す」ボタンをクリック		背景、差し色がデフォルトに戻る	○
TC-13	タスク完全削除	削除済みタスク一覧でタスクを選択し、「完全削除」ボタンをクリック	完全削除するタイトル: 買い物リスト	タスクが完全削除され、タスク一覧に表示されなくなる	○



# 操作手順書

## スタート画面

### タスク管理

①

☒ ライトモード      差し色を選択      

☐ ダークモード

②

新しいカテゴリ:        検索

③

タイトル: (例: 完了タスクの確認)

説明: (例: タスクの詳細を記述)

優先度 (例: 低, 中, 高)      締切      カテゴリ

低      年 / 月 / 日 --:--      趣味

### Tasks

④

タイトル	優先度	期限	カテゴリ
------	-----	----	------

⑤

- ① 背景・差し色の変更
- ② 新しいカテゴリの追加
- ③ 新規タスク入力画面
- ④ タスクリスト
- ⑤ 削除済みタスクへ

## ① 背景・差し色の変更



🎨 をクリックすると背景が変更します




が出てくるのでお好みの色に変更してください。

背景をダークモード、差し色をレッドにしたスタート画面



## ② 新しいカテゴリの追加

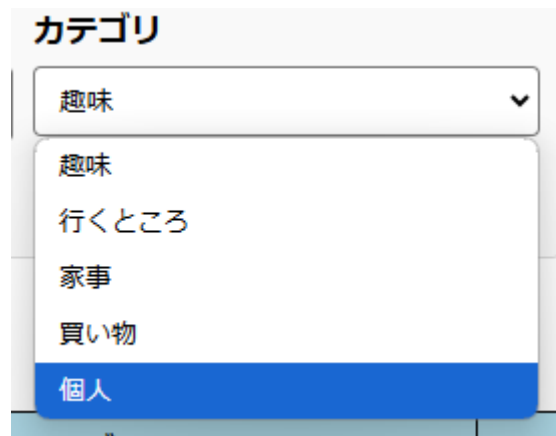


- 「新しいカテゴリ:」の横のフォームに追加したいカテゴリの名前を入力し、「追加」ボタンをクリックしてください。

※ 一度登録のしたことあるカテゴリは登録できません

- 新しいカテゴリを追加後、タスク入力フォームの「カテゴリ」に追加したカテゴリが登録されています。

※ 例)「個人」



## ③ 新規タスク入力画面



- ①タイトル入力フォーム:追加したいタスクのタイトルを入力してください  
※必ず入力してください  
例)買い物
  - ②説明入力フォーム:追加したいタスクの説明を入力してください。  
※必ず入力してください  
例)スーパーで買い物
  - ③優先度:プルダウンで「低」「中」「高」を選択できるようになっています  
※デフォルトで「低」に設定してあります  
例)中
  - ④締切入力フォーム:締め切りの日時がある場合入力してください。  
例)2024-12-20
  - ⑤カテゴリ:登録してあるカテゴリを選択してください。  
※必ず入力してください  
例)個人
- タスクを追加する場合は、「タスクを追加」ボタンをクリック  
「クリア」ボタンをクリックすると入力内容をクリアすることもできます。

#### ④ タスクリスト

##### タスクリスト

タイトル		優先度	期限	カテゴリー	
買い物	詳細	中	2024/12/16 00:00	個人	完了にする 削除

#### ③新規タスク入力画面 の例で追加したタスクリスト

「詳細」のボタンをクリックすると、例で入力した、タスクの詳細を確認できます。

×

買い物

スーパーで買い物

タスクの編集

タイトル:  
買い物

説明:  
スーパーで買い物

優先度:  
中

×

買い物

スーパーで買い物

優先度:  
中

締切:  
2024/12/16 00:00

カテゴリー:  
個人

編集

ポップアップ表示の詳細画面では、タスクの「編集」も行えます。

例)タイトル:買い物リスト

説明:スーパーで必要なもの

編集後 →

×

買い物リスト

スーパーで必要なもの

#### タスクの完了

タスクを完了した場合、「完了にする」ボタンをクリックすると、完了したタスク1行がグレーアウトし、タスクの完了/未完了を確認することができます。

##### タスクリスト

タイトル		優先度	期限	カテゴリー	
買い物リスト	詳細	中	2024/12/16 00:00	個人	未完了にする 削除
薬局	詳細	中		買い物	完了にする 削除

#### タスクの削除

タスクを削除したい場合、「削除」ボタンをクリックします。「削除」ボタンを押されたタスクはタスクリストから削除されます。

タス

買い物リストが削除された

タイトル		優先度	期限	カテゴリー	
薬局	詳細	中		買い物	完了にする 削除

## ⑤ 削除済みタスク

「削除済みタスクを表示」ボタンをクリックすると削除されたタスクを確認できます。

### 削除済みタスク

● ライトモード  
○ ダークモード

差し色を選択:

デフォルトに戻す

①

タスク一覧に戻る

②

タイトル	詳細	優先度	期限	カテゴリー	
買い物リスト	スーパーで必要なもの	中	2024/12/16 00:00	個人	<div style="display: flex; gap: 5px;"> <span>復元</span> <span>完全削除</span> </div>

③

① タスク一覧に戻る: このボタンを押すことでタスク追加、タスクリストの画面に戻ることができます。

② 背景・差し色の変化: タスク一覧と同じくユーザー好みのスタイルに変更できます。

③ 削除されたタスク

→ 🚩 「復元」ボタン: タスクを戻したい場合、このボタンをクリックすると元のタスクリストに復元することができます。

→ 🚩 「完全削除」ボタン: タスクを完全に削除したい場合、このボタンをクリックするとタスクを完全に削除します。



## プロジェクト概要

## 目的と背景

### 目的

- ・ 自分専用のタスク管理システムを開発し、タスクの作成、編集、削除、および閲覧を効率的に行えるようにする。

### 背景

- ・ 以前にPythonで作成したタスク管理アプリケーションを基に、さらに機能を追加し、カスタマイズ性を高めたJavaバージョンを作成することにした。
- ・ ユーザーが日常的にタスク管理を身近に感じられるようにすることを目指しています。

## ターゲットユーザー

## ユーザー

- ・ ユーザー層
  - 一般のユーザー
- ・ ユーザーのニーズ
  - カスタマイズ性の高いインターフェース
  - 効率的なタスク管理
  - 視覚的に魅力的なデザイン

## 機能要件

## 主な機能

01

タスクの作成、編集、削除、復元、完了/未完了の切り替え

02

カテゴリの作成、表示

03

タスクの検索

04

テーマの切り替え、差し色の変更

## システム構成

## 非機能要件



## アーキテクチャ

フロントエンドとバックエンドを統合したシステム

## 技術スタック

• Java, サーブレット技術, PostgreSQL, Tomcat



• フロントエンド技術: HTML5, CSS3, JavaScript



• ツール: Eclipse



## フェーズごとのスケジュール

1

設計フェーズ:

要件定義とシステム設計,  
デザインセックアップの作成,  
データベーススキーマの設計  
(12月7日～12月9日)

2

開発フェーズ:

フロントエンドとバックエンドの構築,  
基本機能の実装,  
テーマ切り替えと差し色変更機能の実装  
(12月10日～12月13日)

3

テストフェーズ:

ユニットテストと統合テストの実施,  
バグの修正, 全体の安定性の確認  
(12月13日)

## 開発スケジュール

## リスクと対策



## リスクと対策

- ・ リスク
  - ・ 異なるデバイスやブラウザで表示や動作に問題が発生する可能性
  - ・ スケジュールの遅延
- ・ 対策
  - ・ 詳細なスケジュール管理と進捗確認
  - ・ 複数のデバイスやブラウザでテストを行い、互換性を確認する。レスポンシブデザインの原則に基づいて設計を行う。



# 品質管理とテスト

## テスト戦略

単体テスト

統合テスト

UIテスト

機能テスト

パフォーマンステスト

セキュリティテスト

## 品質基準

バグなし

直感的で操作しやすいUI

高いパフォーマンス

信頼性

# デモ

# 結論

## 総括

Python版からJava版への移行により、拡張性とカスタマイズ性の高いシステムを実現させようとしたところ欲張りすぎて機能が一時すべて壊れるなど、想定以上に時間がかかりました。

しかし、当初の目的の背景の切り替え、差し色の変更。これができたので悔いはありません。



## 改善点

- ・ 機能拡張
  - ・ 通知機能の強化
  - ・ 並べ替え機能の追加