

【アプリ概要】

HTML・CSS・jQuery を使用して、予定や作業内容を管理する「タスク管理アプリ」を制作します。

【主な機能】

1. タスクの追加

入力欄にタスクを入力し、「追加」ボタンまたは Enter キーで一覧に追加します。

2. 未入力の確認

入力欄が空の場合は追加せず、「タスクを入力してください。」と表示します。

3. タスク一覧

追加したタスクに「完了」ボタンと「削除」ボタンを表示します。

4. 完了状態の切り替え

「完了」ボタンを押すと、タスクに取り消し線を付けます。もう一度押すと未完了状態に戻します。

5. タスクの削除

「削除」ボタンを押すと、選択したタスクを一覧から削除します。

6. 未完了件数の表示

完了していないタスクの件数を表示し、操作のたびに更新します。

7. タスクがない場合の表示

タスクが 0 件の場合は、「登録されているタスクはありません。」と表示します。

8. スマートフォン対応

画面幅が狭い場合でも操作しやすいレイアウトにします。



```
<!DOCTYPE html>
<html lang="ja">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title> タスク管理アプリ </title>
  <link rel="stylesheet" href="style.css">
</head>
<body>
  <main class="task-app">
    <h1> タスク管理 </h1>

    <div class="task-form">
      <label for="taskInput" class="visually-hidden"> 新しいタスク </label>
      <input
        type="text"
        id="taskInput"
        placeholder=" タスクを入力してください "
        autocomplete="off"
      >
      <button type="button" id="addButton"> 追加 </button>
    </div>

    <p class="task-count">
      未完了のタスク： <span id="remainingCount">0</span> 件
    </p>

    <p id="emptyMessage"> 登録されているタスクはありません。 </p>
    <ul id="taskList"></ul>
  </main>

  <script src="https://code.jquery.com/jquery-3.7.1.min.js"></script>
  <script src="script.js"></script>
</body>
</html>
```

タスク管理 APP | CSS (style.css)

```
*{
  box-sizing: border-box;
}

body {
  margin: 0;
  padding: 40px 20px;
  color: #333;
  background-color: #f3f5f7;
  font-family: Arial, "Hiragino Kaku Gothic ProN", "Yu Gothic", sans-serif;
}

button,
input {
  font: inherit;
}

.task-app {
  width: 100%;
  max-width: 640px;
  margin: 0 auto;
  padding: 32px;
  background-color: #fff;
  border-radius: 10px;
  box-shadow: 0 4px 16px rgba(0, 0, 0, 0.1);
}

h1 {
  margin: 0 0 24px;
  text-align: center;
  font-size: 28px;
}

.task-form {
  display: flex;
  gap: 10px;
}

#taskInput {
  flex: 1;
  min-width: 0;
  padding: 12px;
  border: 1px solid #bbb;
  border-radius: 5px;
}

#taskInput:focus {
  border-color: #1976d2;
  outline: 2px solid rgba(25, 118, 210, 0.2);
}

button {
  padding: 10px 16px;
  color: #fff;
  border: 0;
  border-radius: 5px;
  cursor: pointer;
}

#addButton {
  background-color: #1976d2;
}

#addButton:hover {
  background-color: #125ea8;
}

.task-count {
  margin: 24px 0 14px;
  font-weight: bold;
}

#emptyMessage {
  margin: 24px 0 0;
  color: #777;

  text-align: center;
}

#taskList {
  margin: 0;
  padding: 0;
  list-style: none;
}

.task-item {
  display: flex;
  align-items: center;
  gap: 8px;
  margin-top: 10px;
  padding: 12px;
  background-color: #f7f8fa;
  border: 1px solid #ddd;
  border-radius: 6px;
}

.task-text {
  flex: 1;
  overflow-wrap: anywhere;
}

.task-item.completed .task-text {
  color: #888;
  text-decoration: line-through;
}

.complete-button {
  background-color: #2e7d32;
}

.task-item.completed .complete-button {
  background-color: #757575;
}

.delete-button {
  background-color: #d32f2f;
}

.visually-hidden {
  position: absolute;
  width: 1px;
  height: 1px;
  padding: 0;
  margin: -1px;
  overflow: hidden;
  clip: rect(0, 0, 0, 0);
  white-space: nowrap;
  border: 0;
}

@media screen and (max-width: 520px) {
  body {
    padding: 20px 12px;
  }

  .task-app {
    padding: 22px 16px;
  }

  .task-item {
    flex-wrap: wrap;
  }

  .task-text {
    flex-basis: 100%;
  }
}
```

```
$(function () {  
  // 未完了のタスク件数と空欄メッセージを更新する  
  function updateTaskStatus() {  
    const remainingCount = $("#taskList .task-item").not(".completed").length;  
    const taskCount = 問 2-10 ;  
  
    $("#remainingCount").text(remainingCount);  
  
    if (taskCount === 0) {  
      $("#emptyMessage").show();  
    } else {  
      $("#emptyMessage").hide();  
    }  
  }  
}  
  
// 入力された内容から新しいタスクを作成する  
function addTask() {  
  const taskText = 問 2-2 ;  
  
  問 2-3 {  
    alert(" タスクを入力してください。");  
    $("#taskInput").focus();  
    return;  
  }  
  
  const $taskItem = 問 2-4 ;  
  const $taskText = $("").addClass("task-text").text(taskText);  
  const $completeButton = $("")  
    .attr("type", "button")  
    .addClass("complete-button")  
    .text(" 完了 ");  
  const $deleteButton = $("")  
    .attr("type", "button")  
    .addClass("delete-button")  
    .text(" 削除 ");  
  
  $taskItem.append($taskText, $completeButton, $deleteButton);  
  問 2-5 ;  
  
  $("#taskInput").val("").focus();  
  updateTaskStatus();  
}
```

// 追加ボタンをクリックしたとき

```
問 2-1 {  
  addTask();  
};
```

// 入力欄で Enter キーを押したとき

```
$("#taskInput").on("keydown", function (event) {  
  if (event.key === "Enter") {  
    addTask();  
  }  
});
```

// 完了ボタンをクリックしたとき

```
問 2-6 {  
  const $taskItem = 問 2-7 ;
```

```
  問 2-8 ;
```

```
  if ($taskItem.hasClass("completed")) {  
    $(this).text(" 未完了に戻す ");  
  } else {  
    $(this).text(" 完了 ");  
  }
```

```
  updateTaskStatus();  
};
```

// 削除ボタンをクリックしたとき

```
$("#taskList").on("click", ".delete-button", function () {
```

```
  問 2-9  
  updateTaskStatus();  
});
```

```
  updateTaskStatus();  
});
```