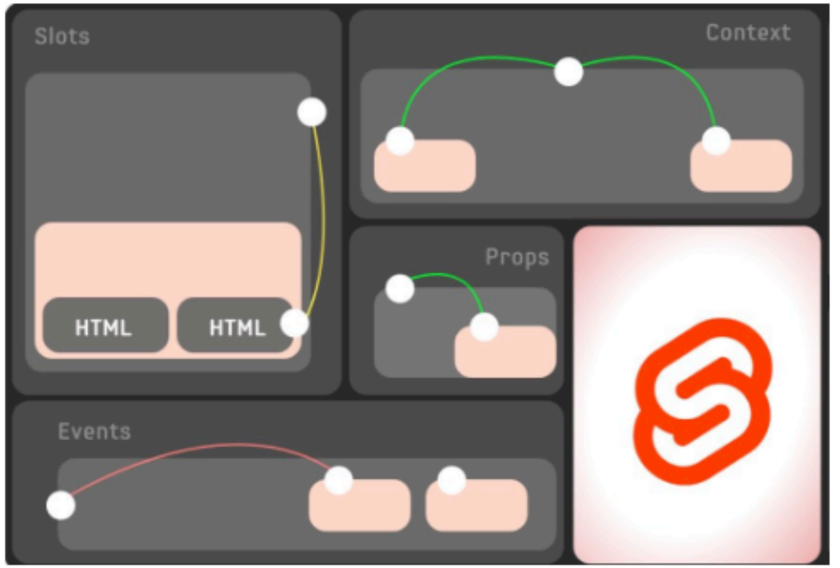
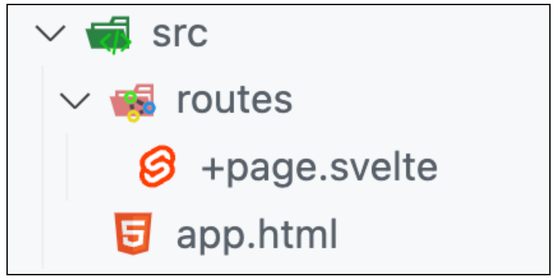
Svelte Components

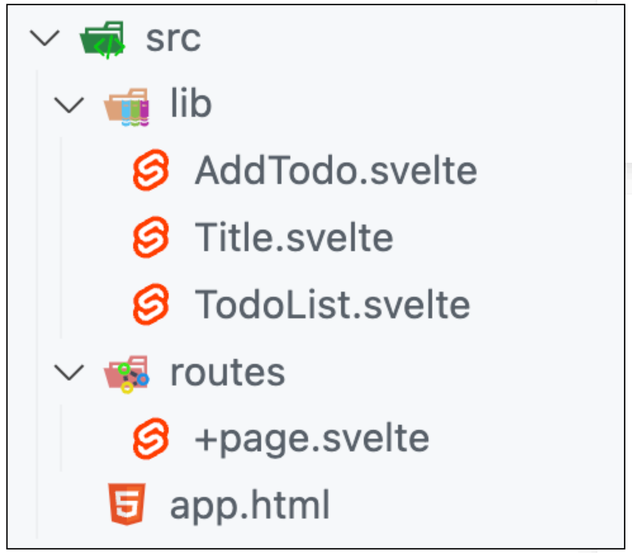


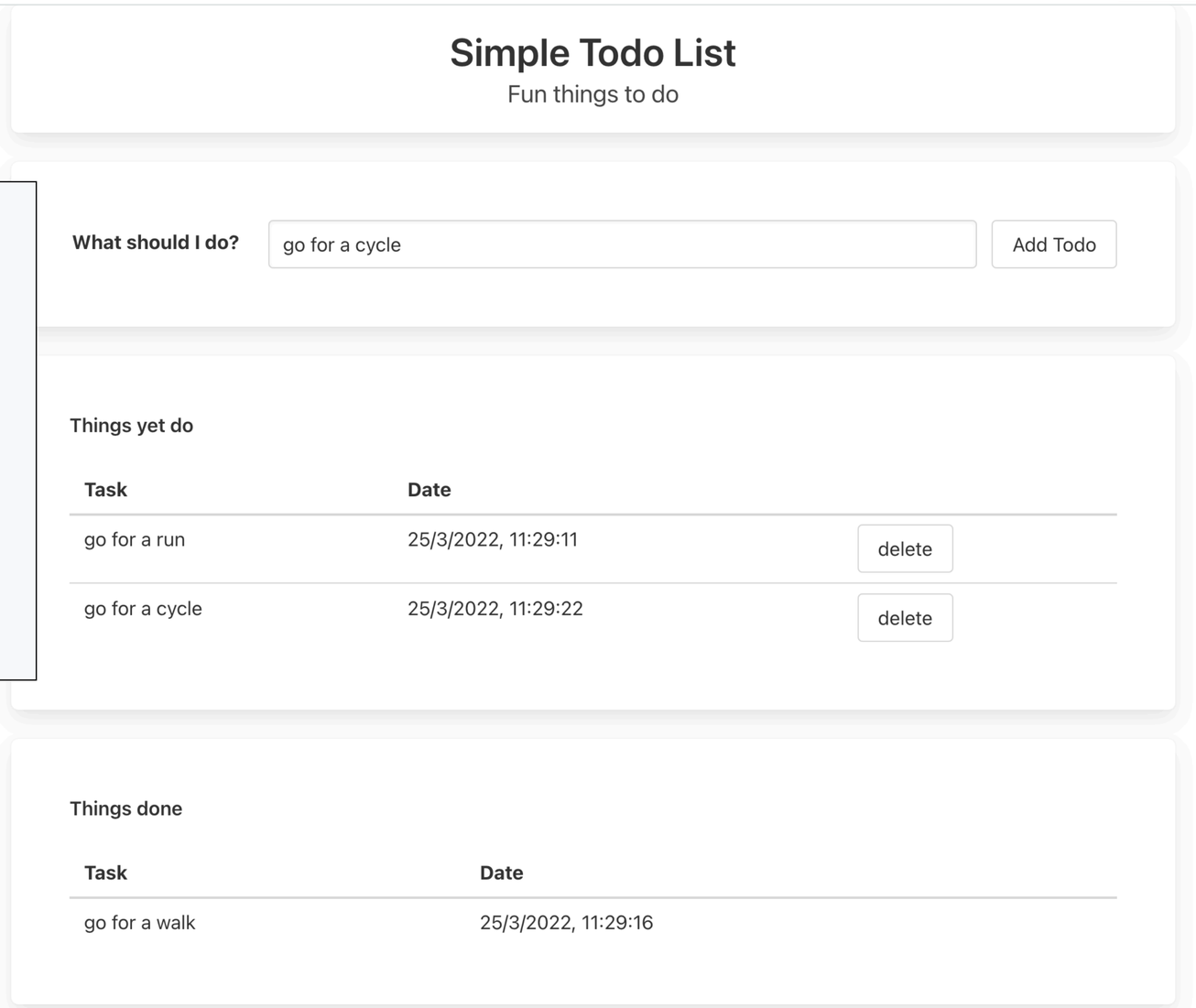
Creating and assembling
Svelte Components

Svelte Components

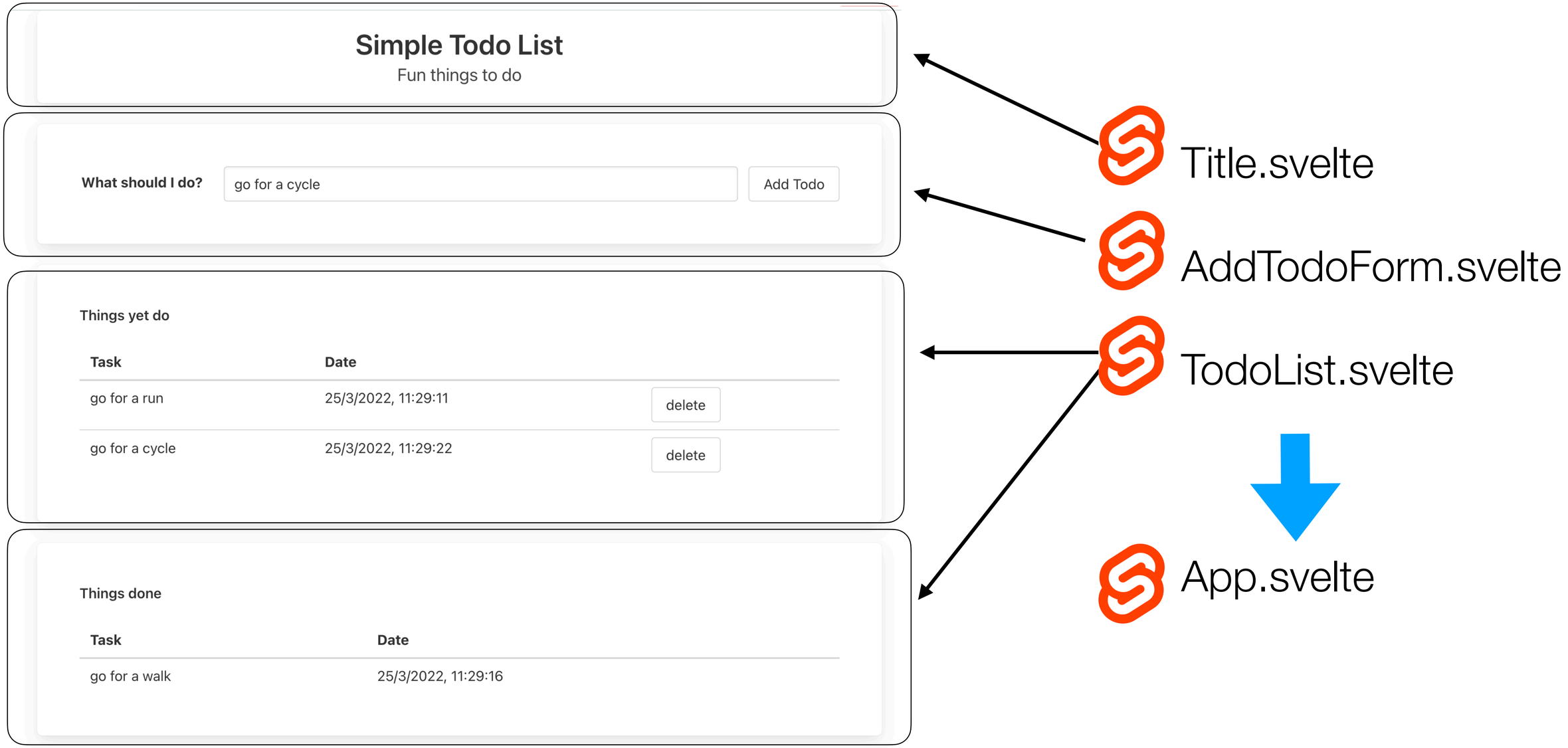


Factor out into (reusable) Svelte components

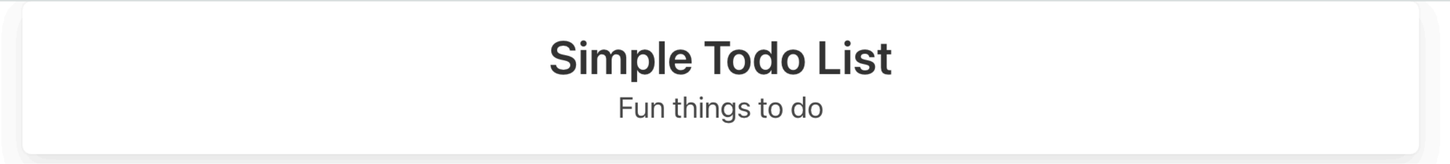




Svelte Components



Title.svelte (1)




```
<div class="box has-text-centered">
  <div class="title"> Simple Todo List</div>
  <div class="subtitle">Fun things to do</div>
</div>
```

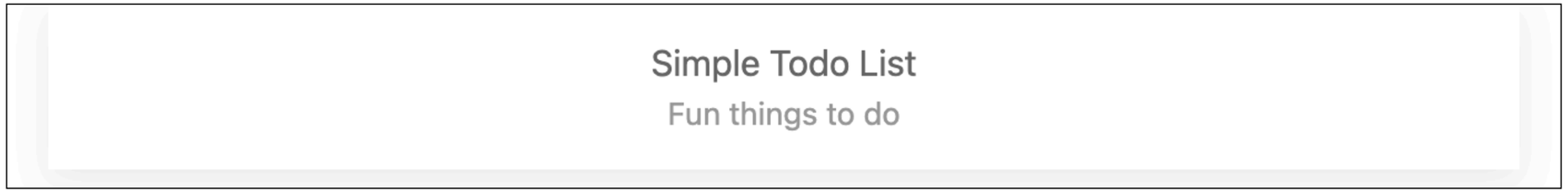
routes/App.svelte

```
import Title from "$lib/Title.svelte";
```

```
<div class="container">
  <Title />
  ...
```

- Simplest possible component
- No parameters or shared features of any kind
- Simply import and use the component as new type of element

Title.svelte (2)




```svelte
<script>
  let { title = "Simple Todo List", subtitle = "Fun things to do" } = $props();
</script>

<div class="box has-text-centered">
  <div class="title">{title}</div>
  <div class="subtitle">{subtitle}</div>
</div>
```

- Export the properties we expect to receive parameters for
- Use these properties in the UX, enclosed in {}
- Pass properties to the component

routes/App.svelte

```svelte
import Title from "$lib/Title.svelte";
```

```svelte
<Title title="Todo Components" subtitle="A new approach" />
```

TodoList.svelte (1)

```svelte
<script>
  let { todoItems = [], deleteTodo } = $props();
</script>

<div class="section box">
  <div class="title is-6">Things yet do</div>
  <table class="table is-fullwidth">
    <thead>
      <tr>
        <th>Task</th>
        <th>Date</th>
      </tr>
    </thead>
    <tbody>
      {#each todoItems as todo}
        <tr>
          <td>{todo.text}</td>
          <td>{todo.date}</td>
          <td><button onclick={() => deleteTodo(todo.id)} class="button">delete</button></td>
        </tr>
      {/each}
    </tbody>
  </table>
</div>
```

Task	Date	
go for a run	25/3/2022, 17:36:17	delete
go for a walk	25/3/2022, 17:36:22	delete
go for a cycle	25/3/2022, 17:36:27	delete

- The properties we expect to receive
- Use these property in the UX, enclosed in {}
- Pass values for property when using the component

routes/+page.svelte

```svelte
import TodoList from "$lib/TodoList.svelte";
```

```svelte
<TodoList {todoItems} {deleteTodo} />
```

+page.svelte (1)

```svelte
<script>
  import { v4 as uuidv4 } from "uuid";
  import Title from "$lib/Title.svelte";
  import TodoList from "$lib/TodoList.svelte";

  let todoText = $state("");
  let todoItems = $state([]);
  let doneItems = $state([]);

  function addTodo() {
    const todo = {
      text: todoText,
      date: new Date().toLocaleString("en-IE"),
      id: uuidv4()
    };
    todoItems.push(todo);
    todoText = "";
  }

  function deleteTodo(id) {
    doneItems.push(todoItems.find((todo) => todo.id === id));
    todoItems = todoItems.filter((todo) => todo.id !== id);
  }
</script>
```

Task	Date	
go for a run	25/3/2022, 17:36:17	delete
go for a walk	25/3/2022, 17:36:22	delete
go for a cycle	25/3/2022, 17:36:27	delete

- Import the components we need
- Declare the application state - the todo text, todo list and done list
- Event handlers to update the state

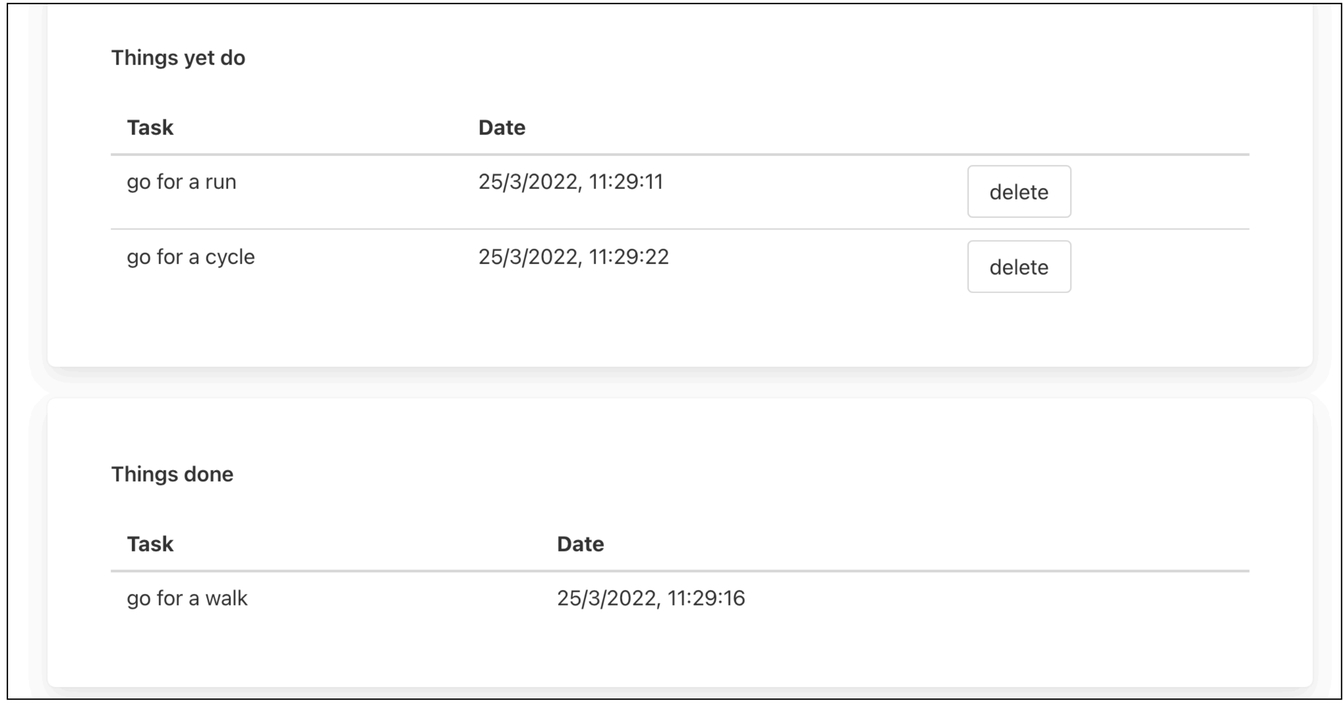
+page.svelte (2)

```svelte
<div class="container">
  <Title title="Todo Components" subtitle="A new approach" />
  <div class="section box">
    <div class="field is-horizontal">
      <div class="field-label is-normal">
        <label for="todo" class="label">What should I do?</label>
      </div>
      <div class="field-body">
        <div class="field">
          <p class="control">
            <input
              bind:value={todoText}
              id="todo"
              class="input"
              type="text"
              placeholder="Type something..."
            />
          </p>
        </div>
        <button onclick={addTodo} class="button">Add Todo</button>
      </div>
    </div>
  </div>

  <TodoList {todoItems} {deleteTodo} />
```

- Input component to accept todo text, bound to a javascript object
- Add button press event handler
- Display the todo list + pass the deleteTodo event handler

+page.svelte (2)

```
<div class="container">
  <Title title="Todo Components" subtitle="A new approach" />
  <div class="section box">
    <div class="field is-horizontal">
      <div class="field-label is-normal">
        <label for="todo" class="label">What should I do?</label>
      </div>
      <div class="field-body">
        <div class="field">
          <p class="control">
            <input
              bind:value={todoText}
              id="todo"
              class="input"
              type="text"
              placeholder="Type something..."
            />
          </p>
        </div>
        <button onclick={addTodo} class="button">Add Todo</button>
      </div>
    </div>
  </div>
  <TodoList {todoItems} {deleteTodo} />
  <TodoList todoItems={doneItems} />
</div>
```

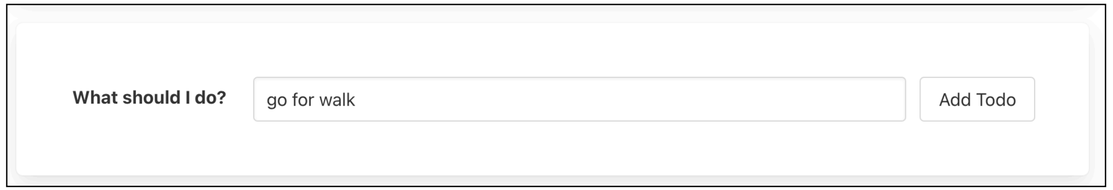


- TodoList component used again for “done” items

AddTodoForm.svelte

lib/AddTodo.svelte

```svelte
<script>
  let { addTodo, todoText = } = $props();
</script>

<div class="section box">
  <div class="field is-horizontal">
    <div class="field-label is-normal">
      <label for="todo" class="label">What should I do?</label>
    </div>
    <div class="field-body">
      <div class="field">
        <p class="control">
          <input
            bind:value={todoText}
            id="todo"
            class="input"
            type="text"
            placeholder="Type something..."
          />
        </p>
      </div>
      <button onclick={() => addTodo()} class="button">Add Todo</button>
    </div>
  </div>
</div>
```



- Factor out AddTodoForm to separate component
- Pass this to the addTodo function todoText as a property
- However, this will not work unless we make the todo time “bindable”:

```svelte
let { addTodo, todoText = $bindable("") } = $props();
```

- Value in todoText will not be reflect in the parent component

routes/+page.svelte

Import components

Application ‘state’

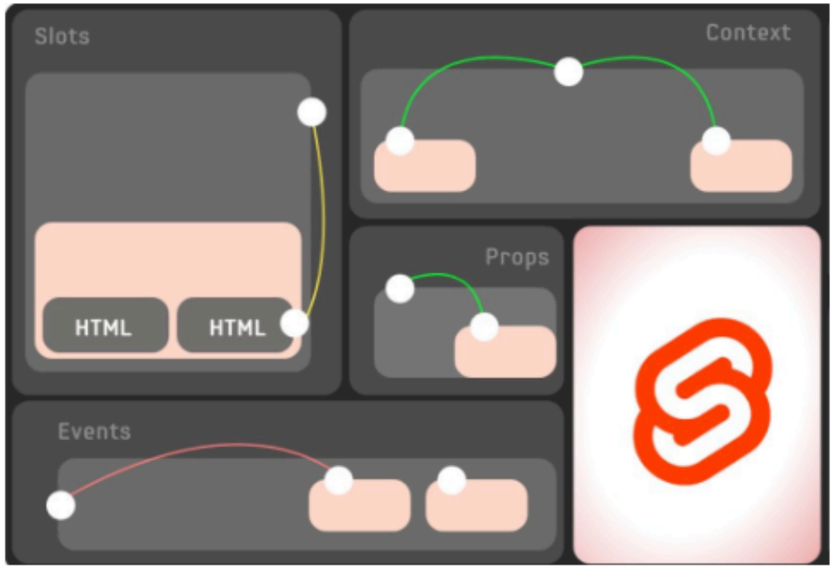
Methods to update state.

UX

routes/+page.svelte

```svelte
<script>
  import { v4 as uuidv4 } from "uuid";
  import Title from "$lib/Title.svelte";
  import TodoList from "$lib/TodoList.svelte";
  import AddTodo from "$lib/AddTodo.svelte";

  let todoText = $state("");
  let todoItems = $state([]);
  let doneItems = $state([]);

  function addTodo() {
    const todo = {
      text: todoText,
      date: new Date().toLocaleString("en-IE"),
      id: uuidv4()
    };
    todoItems.push(todo);
    todoText = "";
  }

  function deleteTodo(id) {
    doneItems.push(todoItems.find((todo) => todo.id === id));
    todoItems = todoItems.filter((todo) => todo.id !== id);
  }
</script>

<div class="container">
  <Title title="Todo Components" subtitle="A new approach" />
  <AddTodo {addTodo} bind:todoText />
  <TodoList {todoItems} {deleteTodo} />
  <TodoList todoItems={doneItems} />
</div>
```

Svelte Components



Creating and assembling
Svelte Components