3

<template> tag

packages

@ember/component	*Input, Textarea*
@ember/helper	*array, concat, fn, get, hash, uniqueId*
@ember/modifier	*on*
@ember/routing	*LinkTo*
@ember/template	*htmlSafe*
@embroider/util	*ensureSafeComponent*

3a

components

my-addon: src/components/hello.hbs

```
<div
  class={{local
    this.styles
    "container"
    (if this.someCondition (array "hide" "after-3-sec"))
  }}
>
  {{t "hello.message" name=@name}}
</div>
```

my-addon: src/components/hello.ts

```ts
import Component from '@glimmer/component';

import styles from './hello.css';

export default class Hello extends Component<HelloSignature> {
  styles = styles;

  get someCondition(): boolean {
    return true;
  }
}
```

my-addon: src/components/hello.ts

```ts
import Component from '@glimmer/component';

import styles from './hello.css';

export default class Hello extends Component<HelloSignature> {
  styles = styles;

  get someCondition(): boolean {
    return true;
  }
}
```

my-addon: src/components/hello.gts

```gts
import Component from '@glimmer/component';

import styles from './hello.css';

export default class Hello extends Component<HelloSignature> {
  styles = styles;

  get someCondition(): boolean {
    return true;
  }

  <template>
  </template>
}
```

my-addon: src/components/hello.gts

```gts
import Component from '@glimmer/component';

import styles from './hello.css';

export default class Hello extends Component<HelloSignature> {
  styles = styles;

  get someCondition(): boolean {
    return true;
  }

  <template>
    <div
      class={{local
        this.styles
        "container"
        (if this.someCondition (array "hide" "after-3-sec"))
      }}
    >
      {{t "hello.message" name=@name}}
    </div>
  </template>
}
```

my-addon: src/components/hello.gts

```gts
import { array } from '@ember/helper';
import Component from '@glimmer/component';
import { t } from 'ember-intl';
import { local } from 'embroider-css-modules';

import styles from './hello.css';

export default class Hello extends Component<HelloSignature> {
  styles = styles;

  get someCondition(): boolean {
    return true;
  }

  <template>
    <div
      class={{local
        this.styles
        "container"
        (if this.someCondition (array "hide" "after-3-sec"))
      }}
    >
      {{t "hello.message" name=@name}}
    </div>
  </template>
}
```

my-addon: src/components/hello.gts

```gts
import { array } from '@ember/helper';
import Component from '@glimmer/component';
import { t } from 'ember-intl';
import { local } from 'embroider-css-modules';

import styles from './hello.css';

export default class Hello extends Component<HelloSignature> {
  get someCondition(): boolean {
    return true;
  }

  <template>
    <div
      class={{local
        styles
        "container"
        (if this.someCondition (array "hide" "after-3-sec"))
      }}
    >
      {{t "hello.message" name=@name}}
    </div>
  </template>
}
```

my-addon: src/components/hello.gts

```gts
import type { TOC } from '@ember/component/template-only';
import { t } from 'ember-intl';

import styles from './hello.css';

const Hello: TOC<HelloSignature> = <template>
  <div class={{styles.container}}>
    {{t "hello.message" name=@name}}
  </div>
</template>;

export default Hello;
```

my-addon: src/components/hello.gts

```gts
import type { TOC } from '@ember/component/template-only';
import { t } from 'ember-intl';

import styles from './hello.css';

<template>
  <div class={{styles.container}}>
    {{t "hello.message" name=@name}}
  </div>
</template> satisfies TOC<HelloSignature>;
```

my-addon: src/index.ts

```ts
export { default as Hello } from './components/hello.gts';
```

my-addon: src/template-registry.ts

```ts
import type HelloComponent from './components/hello.gts';

export default interface MyAddonRegistry {
  Hello: typeof HelloComponent;
  hello: typeof HelloComponent;
}
```

3b

routes

my-app: app/templates/index.hbs

```
1  <div class={{this.styles.container}}>
2    <Hello @name={{this.userName}} />
3  </div>
```

my-app: app/templates/index.gts

```gts
<template>
  <div class={{this.styles.container}}>
    <Hello @name={{this.userName}} />
  </div>
</template>
```

my-app: app/templates/index.gts

```gts
import { Hello } from 'my-addon';

<template>
  <div class={{@controller.styles.container}}>
    <Hello @name={{@controller.userName}} />
  </div>
</template>
```

my-app: app/templates/index.gts

```gts
import type { TOC } from '@ember/component/template-only';
import { Hello } from 'my-addon';
import type IndexController from 'my-app/controllers/index';

interface IndexSignature {
  Args: {
    controller: IndexController;
    model: unknown;
  };
}

<template>
  <div class={{@controller.styles.container}}>
    <Hello @name={{@controller.userName}} />
  </div>
</template> satisfies TOC<IndexSignature>;
```

my-app: app/templates/index.gts

```gts
import type { TOC } from '@ember/component/template-only';
import { Hello } from 'my-addon';
import type IndexController from 'my-app/controllers/index';

import styles from './index.css';

interface IndexSignature {
  Args: {
    controller: IndexController;
    model: unknown;
  };
}

<template>
  <div class={{styles.container}}>
    <Hello @name={{@controller.userName}} />
  </div>
</template> satisfies TOC<IndexSignature>;
```

3c

tests

my-app: tests/integration/components/hello-test.ts

```ts
import {
  render,
  type TestContext as BaseTestContext,
} from '@ember/test-helpers';
import { hbs } from 'ember-cli-htmlbars';
import { setupRenderingTest } from 'my-app/tests/helpers';
import { module, test } from 'qunit';

interface TestContext extends BaseTestContext {
  userName: string;
}

module('Integration | Component | hello', function (hooks) {
  setupRenderingTest(hooks);

  hooks.beforeEach(function (this: TestContext) {
    this.userName = 'Zoey';
  });

  test('it renders', async function (this: TestContext, assert) {
    await render<TestContext>(
      hbs`
        <Hello @name={{this.userName}} />
      `,
    );

    assert.dom().hasText('Hello, Zoey!');
  });
});
```

my-app: tests/integration/components/hello-test.gts

```gts
import {
  render,
  type TestContext as BaseTestContext,
} from '@ember/test-helpers';
import { Hello } from 'my-addon';
import { setupRenderingTest } from 'my-app/tests/helpers';
import { module, test } from 'qunit';

interface TestContext extends BaseTestContext {
  userName: string;
}

module('Integration | Component | hello', function (hooks) {
  setupRenderingTest(hooks);

  hooks.beforeEach(function (this: TestContext) {
    this.userName = 'Zoey';
  });

  test('it renders', async function (this: TestContext, assert) {
    await render(
      <template>
        <Hello @name={{this.userName}} />
      </template>,
    );

    assert.dom().hasText('Hello, Zoey!');
  });
});
```

my-app: tests/integration/components/hello-test.gts

```gts
import {
  render,
  type TestContext as BaseTestContext,
} from '@ember/test-helpers';
import { Hello } from 'my-addon';
import { setupRenderingTest } from 'my-app/tests/helpers';
import { module, test } from 'qunit';

interface TestContext extends BaseTestContext {
  userName: string;
}

module('Integration | Component | hello', function (hooks) {
  setupRenderingTest(hooks);

  hooks.beforeEach(function (this: TestContext) {
    this.userName = 'Zoey';
  });

  test('it renders', async function (this: TestContext, assert) {
    const self = this;

    await render(
      <template>
        <Hello @name={{self.userName}} />
      </template>,
    );

    assert.dom().hasText('Hello, Zoey!');
  });
});
```

my-app: tests/integration/components/hello-test.gts

```gts
import {
  render,
  type TestContext as BaseTestContext,
} from '@ember/test-helpers';
import { Hello } from 'my-addon';
import { setupRenderingTest } from 'my-app/tests/helpers';
import { module, test } from 'qunit';

interface TestContext extends BaseTestContext {
  userName: string;
}

module('Integration | Component | hello', function (hooks) {
  setupRenderingTest(hooks);

  hooks.beforeEach(function (this: TestContext) {
    this.userName = 'Zoey';
  });

  test('it renders', async function (this: TestContext, assert) {
    const { userName } = this;

    await render(
      <template>
        <Hello @name={{userName}} />
      </template>,
    );

    assert.dom().hasText('Hello, Zoey!');
  });
});
```

my-app: tests/integration/components/hello-test.gts

```gts
import { render } from '@ember/test-helpers';
import { Hello } from 'my-addon';
import { setupRenderingTest } from 'my-app/tests/helpers';
import { module, test } from 'qunit';

module('Integration | Component | hello', function (hooks) {
  setupRenderingTest(hooks);

  const userName = 'Zoey';

  test('it renders', async function (assert) {
    await render(
      <template>
        <Hello @name={{userName}} />
      </template>,
    );

    assert.dom().hasText('Hello, Zoey!');
  });
});
```

my-app: tests/integration/components/hello-test.gts

```gts
import { render } from '@ember/test-helpers';
import { Hello } from 'my-addon';
import { setupRenderingTest } from 'my-app/tests/helpers';
import { module, test } from 'qunit';

module('Integration | Component | hello', function (hooks) {
  setupRenderingTest(hooks);

  test('it renders', async function (assert) {
    const userName = 'Zoey';

    await render(
      <template>
        <Hello @name={{userName}} />
      </template>,
    );

    assert.dom().hasText('Hello, Zoey!');
  });
});
```