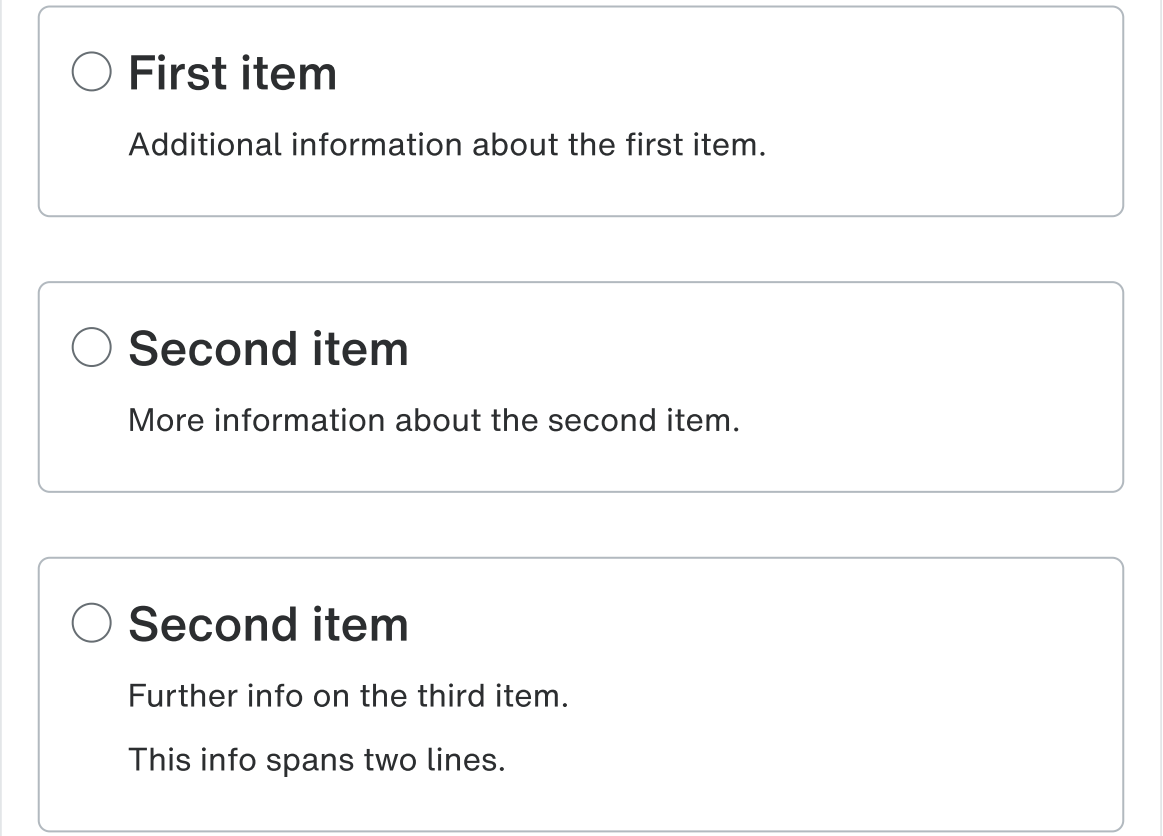
TELUS Component Library

RadioCardGroup

Multi-Platform Component | Figma UI KIT




```
<RadioCardGroup
  items={[
    {
      title: 'First item',
      content: <Typography>Additional information about the
first item.</Typography>
    },
    {
      title: 'Second item',
      content: <Typography>More information about the second
item.</Typography>
    },
    {
      title: 'Second item',
      content: (
        <>
```

```
          <Typography>Further info on the third item.
</Typography>
          <Typography>This info spans two lines.</Typography>
        </>
      )
    }
  ]}
/>
```

Introduction

A RadioCardGroup renders a group of RadioCards, which display information with a title and separate content section like a card, but may be selected like a radio button.

Follow the appropriate instructions to add this component in to your app.

Guidance

Use RadioCardGroup where the user needs to select one item from multiple options, and where those options need descriptive content beyond a simple single line of text. For example, for choosing between named options where additional content beyond the name is needed to make a choice.

> ⚠ CAUTION
>
> To ensure proper functionality when using RadioCardGroup in a React Native application, it is important to wrap it with a `ScrollView` component. This not only ensures that the component renders correctly but also enables users to scroll through the list of RadioCards seamlessly, especially when the content extends beyond the available screen space.

Alternatives

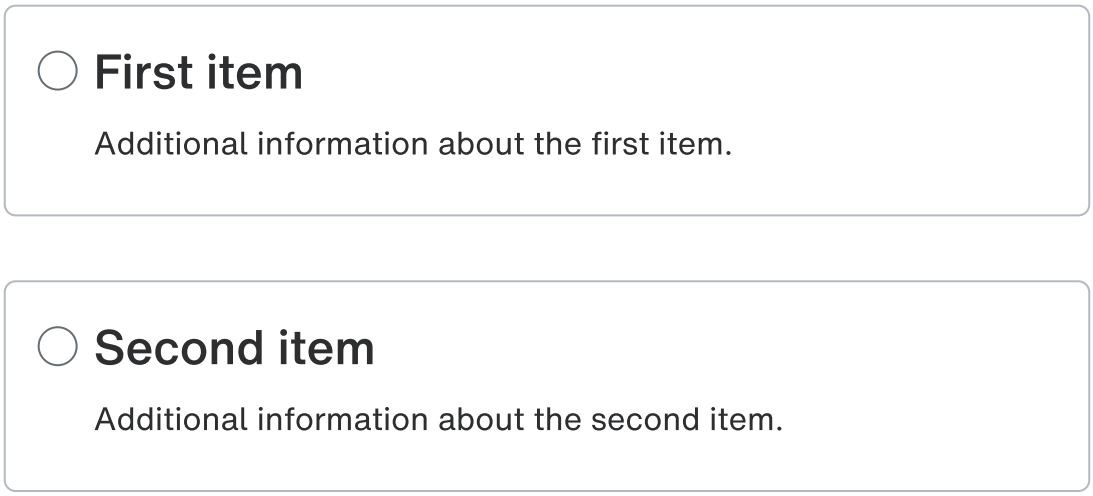
- If the options are fairly simple and can be described in a short phrase, consider Radio
- If the options require additional information but it is supplementary or of interest to only a minority of users rather than essential information for a

majority, consider Radio with a Tooltip
- If the options are very simple and can be described in one or two words or numbers, consider ButtonGroup
- If selecting an option simply navigates a user to a new page describing that option, rather than saving a selection in the current page or form, consider PreviewCard or StoryCard (web only)

Items

Use the `items` prop to pass an array of objects describing each RadioCard in the group:

- `title`: main text passed to RadioCard's `title` prop
- `content`: React content passed to RadioCard's `children` prop
- `id`: identifier used to store which RadioCard is selected (uses `title` if undefinded)
- `onChange`: optional function called on selection, in addition to updating the group's selection state




```
<RadioCardGroup
  items={[
    {
      title: 'First item',
      content: <Typography>Additional information about the
first item.</Typography>,
      id: 'first',
      onChange: () => alert('First item selected')
    },
    {
```

```
      title: 'Second item',
      content: <Typography>Additional information about the
second item.</Typography>,
      id: 'second',
      onChange: () => alert('Second item selected')
    }
  ]}
/>
```

Controlled

The selection state may be controlled by a parent by passing `checkedId` and an `onChange` function that updates the parent's state. If the component is controlled in this way, `checkedId` must never be undefined; pass `null` or an empty string if there is no selection.

Currently selected: ""

◯ **First item**

More information here.

◯ **Second item**

More information here.

◯ **Third item**

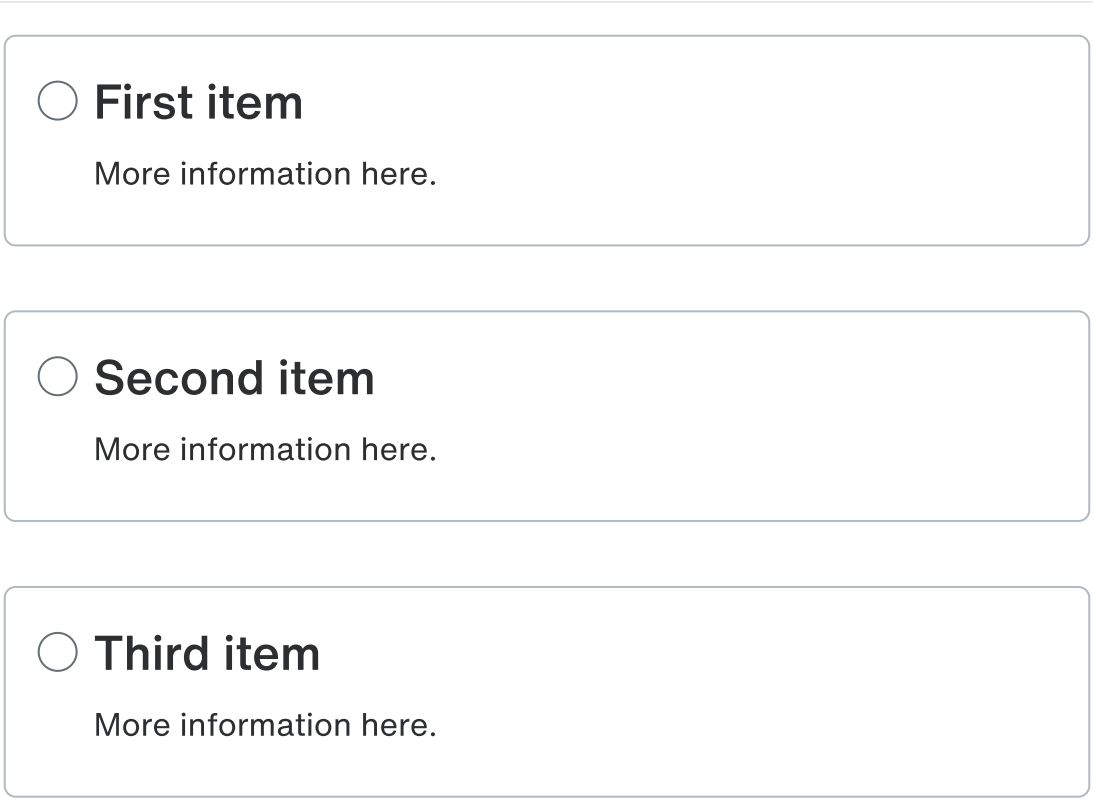
More information here.

```
function ControlledRadioButtonGroup() {
  const [checkedId, setCheckedId] = useState('')
  return (
    <StackView space={3}>
      <Typography>Currently selected: "{checkedId}"</Typography>
      <RadioCardGroup
```

```
        checkedId={checkedId}
        onChange={setCheckedId}
        items={[
          { title: 'First item', content: <Typography>More
information here.</Typography> },
          { title: 'Second item', content: <Typography>More
information here.</Typography> },
          { title: 'Third item', content: <Typography>More
information here.</Typography> }
        ]}
      />
    </StackView>
  )
}
```

Uncontrolled

If `checkedId` is not provided, RadioCardGroup manages its own state. An initial selection may be provided by passing `initialCheckedId`.




```
<RadioCardGroup
  initialCheckedId="Second item"
  items={[
```

```
    { title: 'First item', content: <Typography>More information
here.</Typography> },
    { title: 'Second item', content: <Typography>More
information here.</Typography> },
    { title: 'Third item', content: <Typography>More information
here.</Typography> }
  ]}
/>
```

Use in forms

For web forms, the `name` prop may be used to define the name of the group's `<fieldset>` and input elements, and `legend` may be used to provide a title with `<legend>` type for the fieldset.

Choose a product

◯ **First product**

A truly high quality product.

◯ **Second product**

Another excellent product.

◯ **Third product**

Not as good as the others, to be honest.

```
<RadioCardGroup
  legend="Choose a product"
  name="product-choice"
  items={[
    {
      title: 'First product',
      id: 'x019',
```

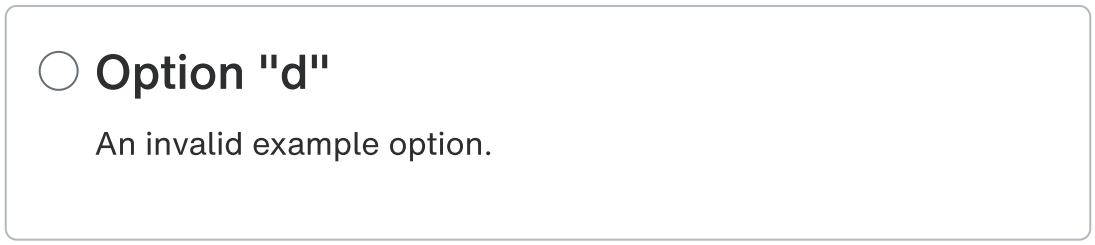
```
      content: <Typography>A truly high quality product.
</Typography>
    },
    {
      title: 'Second product',
      id: 'y022',
      content: <Typography>Another excellent product.
</Typography>
    },
    {
      title: 'Third product',
      id: 'z144',
      content: <Typography>Not as good as the others, to be
honest.</Typography>
    }
  ]}
/>
```

Validation

Validation state may be set by passing 'error' or 'success' to the `validation` prop.

Choose 'a' or 'c'

- ○ **Option "a"**
  A valid example option.
- ○ **Option "b"**
  An invalid example option.
- ○ **Option "c"**
  A valid example option.




```
function ValidationExample() {
  const [validation, setValidation] = useState(null)
  const validate = (id) => setValidation(['a', 'c'].includes(id)
? 'success' : 'error')
  return (
    <RadioCardGroup
      legend="Choose 'a' or 'c'"
      onChange={validate}
      validation={validation}
      feedback={validation && `${validation === 'success' ?
'Valid' : 'Invalid'} item chosen.`}
      items={[
        { title: 'Option "a"', id: 'a', content: <Typography>A
valid example option.</Typography> },
        {
          title: 'Option "b"',
          id: 'b',
          content: <Typography>An invalid example option.
</Typography>
        },
        { title: 'Option "c"', id: 'c', content: <Typography>A
valid example option.</Typography> },
        {
          title: 'Option "d"',
          id: 'd',
          content: <Typography>An invalid example option.
</Typography>
        }
      ]}
    />
  )
}
```

A11y guidelines

RadioCardGroup accepts all the common accessibility props, but also sets accessibility role `'radiogroup'` and controls the accessibility state of children like RadioCard and Feedback based on current state.

RadioCard component accepts all the common accessibility props, but also sets some defaults, including accessibility role `'radio'` and accessibility state that depends on the other props (`checked` , `inactive`) or the internal state in case of uncontrolled RadioCard.

Platform considerations

The component is available on both native platforms and web.

Props

Name	Type	Platform	Default	Description
copy	'en' | 'fr'	standard		Whether the English or French copy will be used (e.g. for accessibility labels).
tokens	tokens	standard		System tokens prop, see tokens for more details
radioCardTokens	custom	standard		Optional theme token overrides for each inner RadioCard component

Name	Type	Platform	Default	Description
variant	variant	standard		System variant prop, see variants for more details
items	arrayOf	standard		Array of objects containing specifics for each RadioCard to be rendered in the group.
legend	string	standard		Main text used to describe this group, used in Fieldset's Legend element.
hint	string	standard		Optional additional text giving more detail to help a user make a choice.
hintPosition	'inline' | 'below'	standard		Position of the hint relative to label. Use `below` to display a larger hint below the label.

Name	Type	Platform	Default	Description
tooltip	string	standard		Optional tooltip text content to include alongside the legend and hint.
validation	'error' | 'success'	standard		Current validation status of the group, passed to the feedback element if there is one.
feedback	string	standard		If provided, a Feedback element is rendered containing this text.
initialCheckedId	string	standard		If provided, the radio card with this id is selected on first render.
checkedId	string	standard		If not undefined, the radio card with this id is selected (or none is selected if

Name	Type	Platform	Default	Description
				`null`), and the element's selection state will be controlled by its parent using the `onChange` function.
onChange	func	standard		Function to call on change in selection state. Is required if the selection state is controlled by a parent using checkedId and the input is not readOnly.
readOnly	bool	standard		If true, the radio cards cannot be selected by the user and simply show their current state.
inactive	bool	standard		If true, the radio card cannot be interacted with, elements are set as

Name	Type	Platform	Default	Description
				`disabled` and if the theme supports `inactive` appearances rules, these are applied.
name	string	standard		On Web, this is passed to the `name` attribute of the fieldset and each radio input.

Tokens

In exceptional circumstances, the following tokens can be passed to this component to override its default styles. **Do not do this unless absolutely necessary.** Read more about overriding styles.

▶ View Tokens

Variants

This component does not have any stylistic variants.

Feedback

- Spotted a problem with this component? Raise an issue on GitHub
- See any existing issues for this component
- Contact the team on slack in #ds-support