Migração do @material-ui-pickers

@material-ui/pickers foi movido para o @material-ui/lab.

> ⚠️ ***Os componentes seletores de data foram reescritos***. *Na maioria dos lugares, a lógica foi reescrita do zero, então não é possível especificar aqui toda a lista de mudanças. Aqui está uma visão geral dos conceitos mais importantes que foram alterados. Se você vai atualizar, o caminho mais fácil pode ser identificar o uso de cada seletor em sua base de código, e reescreva-os um de cada vez. Não se esqueça de executar seus testes depois de cada um!*

Este guia é uma visão geral dos conceitos principais que foram alterados de seletores na v3.2.10.

Instalação

You need to install the `@material-ui/lab` package if it's not already installed. ⚠️ Make sure you have installed the latest version, `"@material-ui/lab": ^5.0.0-alpha.30` or above.

Importações

A versão `keyboard` dos seletores não está mais publicada. Todas as versões dos seletores de dispositivos móveis e desktop implementam a entrada do teclado para acessibilidade.

```
-import { KeyboardDatePicker } from '@material-ui/pickers';
+import DatePicker from '@material-ui/lab/DatePicker';

-<KeyboardDatePicker />
+<DatePicker />
```

Além disso, em vez de fornecer uma propriedade `variant`, eles foram movidos para diferentes importações, significando que seu pacote não incluirá um `Dialog` se você estiver usando apenas o seletor de desktop.

- `<DesktopDatePicker />` – Somente visualização em desktop.
- `<MobileDatePicker />` – Somente visualização em dispositivos móveis.
- `<DatePicker />` – Visualização móvel ou desktop de acordo com a preferência de **pointer** do usuário.
- `<StaticDatePicker />` – A própria visualização do seletor em si, sem um input ou qualquer componente encapsulado.

```
-import { DatePicker } from '@material-ui/pickers';
+import DesktopDatePicker from '@material-ui/lab/DesktopDatePicker';

-<DatePicker variant="inline" />
+<DesktopDatePicker />
```

A mesma convenção se aplica para `TimePicker` – `<DesktopTimePicker>` e `<MobileTimePicker />`.

MuiPickersUtilsProvider

O `MuiPickersUtilsProvider` foi removido em favor do `LocalizationProvider`. Além disso, os seletores não exigem que você instale adaptadores de date-io manualmente. Tudo está incluído no `lab`.

❌ Antes:

```
import AdapterDateFns from '@date-io/date-fns';
import { MuiPickersUtilsProvider } from '@material-ui/pickers';
```

✅ Depois:

```
import AdapterDateFns from '@material-ui/lab/AdapterDateFns';
import LocalizationProvider from '@material-ui/lab/LocalizationProvider';


function App() {
  return (
    <LocalizationProvider dateAdapter={AdapterDateFns}>
      ...
    </LocalizationProvider>
  )
);
```

Renderizando input

Introduzimos uma nova propriedade **requerida**, `renderInput` . Isso simplifica o uso de componentes de entrada de texto sem Material-UI.

```
<DatePicker renderInput={(props) => <TextField {...props} />} />
<TimePicker renderInput={(props) => <TextField {...props} />} />
```

Anteriormente, propriedades eram espalhadas no componentes `<TextField />` . De agora em diante você precisará usar a nova propriedade `renderInput` para fornecer as propriedades:

```
 <DatePicker
- label="Date"
- helperText="Something"
+ renderInput={props => <TextField label="Date" helperText="Something" /> }
/>
```

Gerenciamento do estado

A lógica de gerenciamento de estado/valor para seletores foi reescrita do zero. Seletores agora vão chamar a propriedade `onChange` quando cada visualização do seletor de data estiver sido concluída. O manipulador `onError` esta também totalmente diferente. Verifique a integração dos seus seletores com formulários, porque pode haver alguma questão referente a integração.

Nenhuma máscara necessária

Máscara já não é necessária. Além disso, se a máscara fornecida não for válida, os seletores irão simplesmente ignorar a máscara e permitir entradas arbitrárias.

```
<DatePicker
  mask="mm"
  value={new Date()}
  onChange={console.log}
  renderInput={(props) => (
    <TextField {...props} helperText="invalid mask" />
  )}
/>

<DatePicker
  value={new Date()}
  onChange={console.log}
  renderInput={(props) => (
    <TextField {...props} helperText="valid mask" />
  )}
/>
```

E muito mais

- ````diff <DatePicker
  - format="DD-MMM-YYYY"
  - inputFormat="DD-MMM-YYYY" ``` ````

Há muitas mudanças, tenha cuidado, certifique-se de que seus testes e build passe. No evento em que você tem um uso avançado do seletor de data, provavelmente será mais simples reescrevê-lo.

Por favor, abra um pull request para melhorar o guia se você notar uma oportunidade para fazer isso.