Vitor Amadeu Souza

Introdução ao

React

e

CSS

No ambiente Replit

Parte IV

Agosto de 2024

Direitos reservados por:
Cerne Tecnologia e Treinamento Ltda
Produção: Cerne Tecnologia e Treinamento
E-mail da Empresa: cerne@cerne-tec.com.br
Home Page: www.cerne-tec.com.br.com.br
Atendimento ao Consumidor: sac@cerne-tec.com.br
Contato com o Autor: vitor@cerne-tec.com.br

FEITO NO BRASIL

"Venha também sobre mim a tua benignidade, ó Senhor, e a tua salvação, segundo a tua palavra."

Sl 119:41

Cerne Tecnologia

A Cerne Tecnologia tem uma equipe preparada para desenvolvimento de projetos eletrônicos em diversas áreas: Médica, Entretenimento, Industrial, Robótica, Científica, Automobilística, Aeronáutica, etc. Trabalhamos com tecnologia microcontrolada usando o PIC, ARM, AVR, 8051, dsPIC, PIC24, PIC32 além do Arduino, Raspberry, Beaglebone etc. Desenvolvemos o projeto desde sua concepção até a entrega do produto final, passando pelas etapas de esquema elétrico, protótipo e desenvolvimento de circuito impresso.

Desenvolvemos aplicativos para smartphones/tablets Android, iOS, Blackberry, Windows Phone e no desenvolvimento de softwares a nível PC para plataforma Windows, usando ferramentas como o Visual Basic, C# e C++.

Atuamos na parte de montagem de placas, onde podemos fornecer ambos os serviços de desenvolvimento de projetos e produção ou apenas um destes.

Desenvolvemos esquemas elétricos e layout de PCI, tanto em tecnologia convencional como SMD.

Temos a flexibilidade de customizar um de nossos produtos, de modo a atender a uma necessidade específica do cliente, tornando o custo de desenvolvimento menor se comparado a construção de um projeto desde a sua fase inicial.

Desenvolvemos e fornecemos kits didáticos para diversos microcontroladores além de apostilas, livros e e-books.

Na hora de desenvolver um projeto ou equipar seu laboratório não hesite em nos contatar. Entre em contato conosco através do endereço cerne-tec.com.br para obter mais informações.

cerne-tec.com.br

Sumário

Capítulo I – Metodologia de desenvolvimento **6**

1. Introdução 6

Capítulo II – Programação em React e CSS **7**

1. Testando o flex-grow 7
2. flex-shrink 10
3. Testando o flex-shrink 11
4. flex-basis 14
5. Exemplo com flex-basis 15

Capítulo I
Metodologia de desenvolvimento

1. Introdução

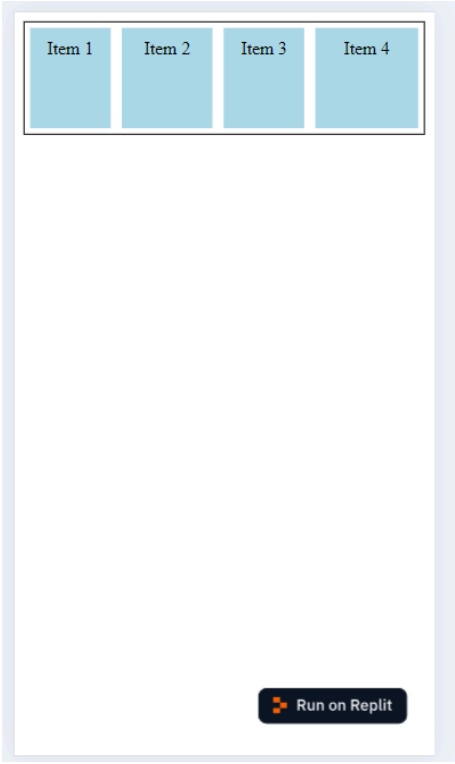
Neste livro, você encontrará uma parte da série sobre o desenvolvimento de projetos utilizando React e CSS. Ao longo desta obra, diversos experimentos serão apresentados segundo o sumário da obra, permitindo o aperfeiçoamento neste framework para a construção de interfaces responsivas juntamente com o CSS.

Capítulo II

Programação em React e CSS

1. Testando o flex-grow

O objetivo do próximo exemplo é adicionar quatro itens a uma div, como o apresentado na próxima figura.



Acompanhe como ficam os arquivos App.css e App.jsx para este exercício.

App.css

```
.container {
  display: flex; /* Define o contêiner como um contêiner flexível */
  border: 1px solid #000; /* Adiciona uma borda preta ao contêiner
*/
  height: 100px; /* Define a altura do contêiner */
}

.item {
  background-color: lightblue; /* azul claro */
  margin: 5px; /* Adiciona uma margem de 5px ao redor dos itens */
  padding: 10px; /*  preenchimento interno de 10px aos itens */
  text-align: center; /* Centraliza o texto dentro do item */
}

.item1 {
  flex-grow: 1; /* Este item crescerá para ocupar espaço extra */
}

.item2 {
  flex-grow: 2; /* Este item crescerá duas vezes mais que item1 */
}

.item3 {
  flex-grow: 1; /* Este item crescerá para ocupar espaço extra */
}

.item4 {
  flex-grow: 3; /* Este item crescerá para ocupar espaço extra */
```

```
}
```

App.jsx

```
import './App.css'

export default function App() {
  return (
    <div className="container">
      <div className="item item1">Item 1</div>
      <div className="item item2">Item 2</div>
      <div className="item item3">Item 3</div>
      <div className="item item4">Item 4</div>
    </div>
  )
}
```

Os códigos fornecidos definem um contêiner flexível que contém quatro itens, cada um com comportamento específico em relação ao espaço que ocupa dentro do contêiner. No arquivo CSS, o contêiner é configurado como um contêiner flexível usando display: flex;, tem uma borda preta de 1 pixel e uma altura de 100 pixels. Cada item dentro do contêiner tem um fundo azul claro, uma margem de 5 pixels, preenchimento interno de 10 pixels, e o texto é centralizado dentro de cada item.

Os itens item1, item2, item3 e item4 utilizam a propriedade flex-grow para definir como cada um deles se expande para ocupar o

espaço disponível no contêiner. O item1 e o item3 têm flex-grow: 1, o que significa que eles crescem para ocupar uma parte do espaço extra. O item2 tem flex-grow: 2, então ele crescerá duas vezes mais que item1 e item3. O item4 tem flex-grow: 3, o que o faz crescer três vezes mais que item1 e item3, e uma vez e meia mais que item2. No arquivo App.jsx, o componente App renderiza esses quatro itens dentro do contêiner, e a distribuição de espaço entre eles é feita de forma dinâmica e proporcional aos valores de flex-grow, resultando em um layout flexível e responsivo.

2. flex-shrink

A propriedade flex-shrink no CSS é usada para definir a capacidade de um item flexível encolher dentro de um contêiner flexível quando o espaço disponível é reduzido. Ela especifica a proporção em que o item deve encolher em relação aos outros itens no contêiner quando há falta de espaço. Por padrão, o valor de flex-shrink é 1, o que significa que os itens irão encolher igualmente para se ajustarem ao espaço disponível. No entanto, se você atribuir valores diferentes de flex-shrink aos itens, aqueles com valores maiores irão encolher mais do que os com valores menores. Por exemplo, se um item tem flex-shrink: 2 e outro tem flex-shrink: 1, o primeiro item encolherá duas vezes mais do que o segundo para se ajustar ao espaço disponível.