

## Selector

Apesar [da documentação \(https://flutter.dev/docs/development/data-and-backend/state-mgmt/simple\)](https://flutter.dev/docs/development/data-and-backend/state-mgmt/simple) explicar sobre [Consumer \(https://pub.dev/documentation/provider/latest/provider/Consumer-class.html\)](https://pub.dev/documentation/provider/latest/provider/Consumer-class.html) e [Provider.of \(https://pub.dev/documentation/provider/latest/provider/Provider.of.html\)](https://pub.dev/documentation/provider/latest/provider/Provider.of.html), existe uma terceira forma de consumir os nossos dados: através do [Selector \(https://pub.dev/documentation/provider/latest/provider/Selector-class.html\)](https://pub.dev/documentation/provider/latest/provider/Selector-class.html) que foi implementada mais recentemente.

Lendo a documentação, percebemos que, no fundo, tanto o Consumer quanto o Selector chamam o Provider.of e buscam otimizar a performance da nossa aplicação de formas diferentes.

O Consumer busca otimizar nossa aplicação granularizando os constantes rebuilds (remontagens) que fazemos na nossa aplicação, fazendo com que a menor parte possível seja remontada. Além disso, também é possível utilizar o Consumer sem um context (diferente de usar o Provider.of diretamente).

Já o Selector, por sua vez, é um equivalente ao Consumer, porém com um filtro específico, onde ele tem valores específicos que ele quer verificar se foram ou não alterados, e não a classe como um todo.