

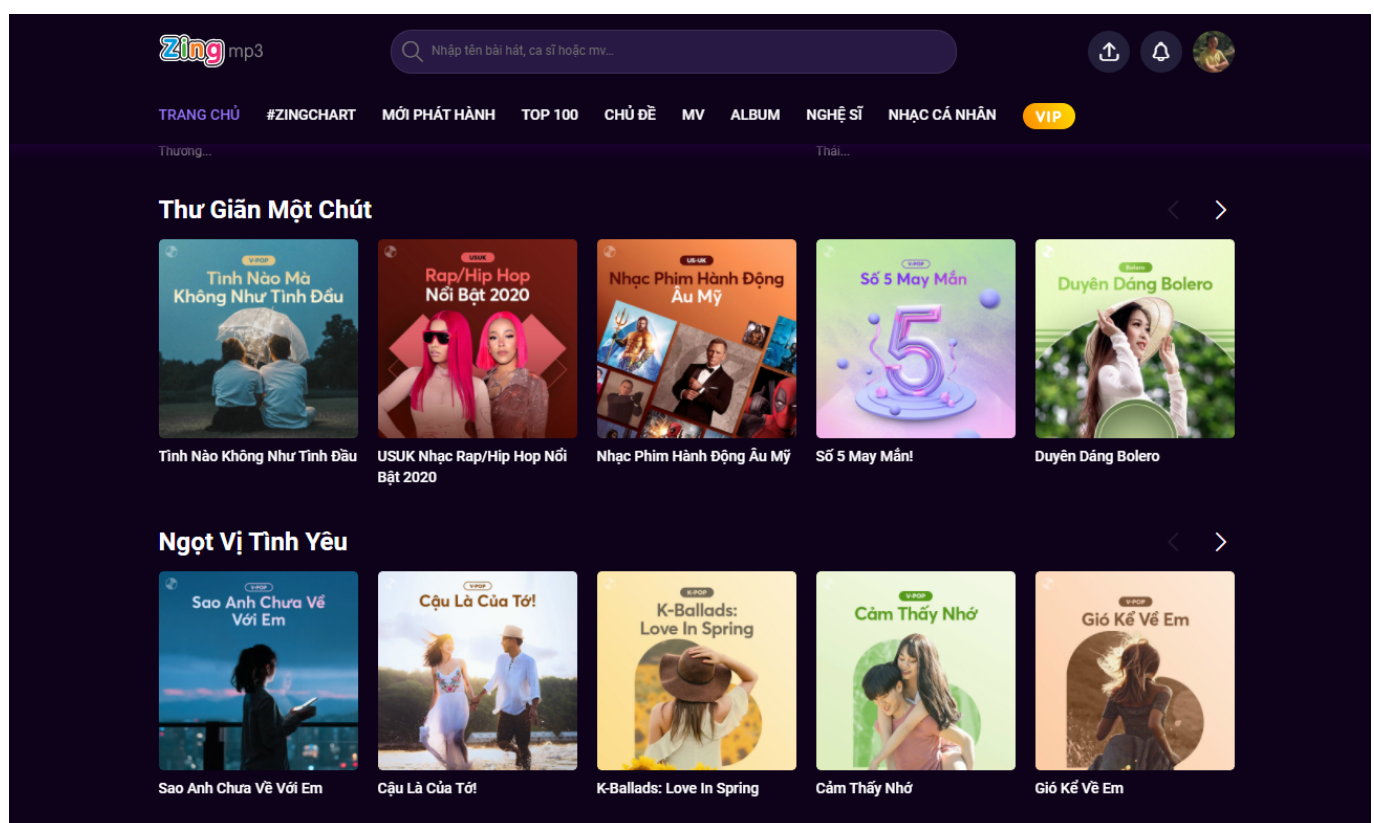
# Props declaration

## Agenda

1. Props là gì?
2. Cách khai báo props sử dụng javascript và typescript?
3. Các loại kiểu dữ liệu cho props
4. Default values cho props

## 1. Props là gì?

- Props là dữ liệu được truyền từ **component cha** xuống **component con**.
- Props là thuộc tính **READ-ONLY**, component con không thay đổi được. Muốn đổi thì nhờ component cha.
- Props giúp tạo ra **sự đa dạng** cho component. Cùng **một component** với **props khác nhau** thì **render ra khác nhau**.



## 2. Cách khai báo props sử dụng javascript và typescript?

### Khai báo props sử dụng Javascript

- **PropTypes** là optional, không bắt buộc phải sử dụng
- Bỏ đi code liên quan tới PropTypes code vẫn chạy bình thường.
- Nhưng nên có **PropTypes** để giúp mình biết được component đó có những props nào mà không cần phải đọc hết code của component đó.

```
import PropTypes from 'prop-types'

function Student(props) {
  return <div>{ props.name }</div>
}

Student.propTypes = {
  name: PropTypes.string.isRequired,
}
```

### Destructuring props

```
import PropTypes from 'prop-types'

function Student({ name }) {
  return <div>{ name }</div>
}

Student.propTypes = {
  name: PropTypes.string.isRequired,
}
```

Với typescript, mình phải khai báo kiểu dữ liệu cho props bằng cách định nghĩa một interface và đặt tên cho nó có suffix là Props.

- Ví dụ về tên: Student**Props**, Item**Props**, StudentList**Props**, ...
- Có bắt buộc dùng **interface** không, em thích **type** được không? --> được, tùy lựa chọn của bạn nhé.
- Bạn có thể search thêm về chủ đề: **interface or type** nhé.

```
interface StudentProps {
  name: string
}

function Student(props: StudentProps) {
  return <div>{ props.name }</div>
}
```

```
interface StudentProps {  
  name: string  
}  
  
function Student({ name }: StudentProps) {  
  return <div>{ name }</div>  
}
```

### 3. Các loại kiểu dữ liệu cho props

- Thêm **dấu chấm hỏi** ngay sau tên thuộc tính nếu là **optional**
- Tham khảo cách định nghĩa props bên dưới.

```
interface Contact {  
  name: string  
  phone: string  
}  
  
interface StudentProps {  
  name: string  
  age: number  
  isHero: boolean  
  address?: string  
  
  hobbyList: string[]  
  primaryContact: Contact  
  onClick: () => string  
}
```

### 4. Default values cho props

Lúc trước mình có dùng **defaultProps** để định nghĩa các giá trị mặc định cho props

```
import PropTypes from 'prop-types'  
  
function Student({ name }) {  
  return <div>{ name }</div>  
}  
  
Student.propTypes = {  
  name: PropTypes.string.isRequired,  
  address: PropTypes.string,  
}  
  
Student.defaultProps = {  
  address: ''  
}
```

Với **function component**, mình có thêm một syntax khác hỗ trợ cho việc default props, nó tên là **default parameters**

```
import PropTypes from 'prop-types'

function Student({ name, address = '' }) {
  return <div>{ name }</div>
}

Student.propTypes = {
  name: PropTypes.string.isRequired,
  address: PropTypes.string,
}
```

Cách viết bên **typescript**

```
interface StudentProps {
  name: string
  address?: string
}

function Student({ name, address = '' }: StudentProps) {
  return <div>{ name }</div>
}
```

## 5. Một số lưu ý

### Nên gom nhóm props

```
interface StudentProps {
  name: string
  age: number
  gender: string
  address?: string
}

function Student({ name, age, gender, address = '' }: StudentProps) {
  return <div>{ name } { age } { gender } {address}</div>
}

function App() {
  return (
    <div>
      <Student name="John" age={10} gender="male" />
    </div>
  )
}
```

--> GOM nhóm props

```
interface Student {
  name: string
  age: number
  gender: string
  address?: string
}

interface StudentProps {
  student: Student
}

function Student({ student }: StudentProps) {
  return <div>{ student.name } { student.address }</div>
}

function App() {
  return (
    <div>
      <Student student={student} />
    </div>
  )
}
```

## Cẩn thận props bị null / undefined

```
interface Student {
  name: string
  age: number
  gender: string
  address?: string
}

interface StudentProps {
  student?: Student
}

// Solution 1: use optional chaining
function Student({ student }: StudentProps) {
  return <div>{ student?.name } { student?.address }</div>
}

// Solution 2: check null/undefined first
function Student({ student }: StudentProps) {
  if (!student) return null


  return <div>{ student.name } { student.address }</div>
}
```

---

## Khoá học ReactJS cho người mới bắt đầu 2021 🍌

- Tác giả: **Hậu Nguyễn** - Founder Easy Frontend
- Khoá học chỉ được published trên Udemy, không thông qua trung gian.
- Khoá học không bán dạng videos upload trên Google Drive hay bất cứ hình thức nào tương tự.
- Khoá học có nhóm discord để hỗ trợ trong quá trình học tập.

 Liên hệ tác giả để được hỗ trợ:

-  Facebook: <https://www.facebook.com/nvhauesmn/>
-  Fanpage: <https://www.facebook.com/learn.easyfrontend>
-  Youtube Channel: <https://www.youtube.com/easyfrontend>