Sunday • 更新于 2022-10-19

上一节 01：前言 03：扩展知识：... 下一节

02：扩展知识：JavaScript与有限自动状态机

我们知道想要实现 `compiler` 第一步是构建 `AST` 对象。那么想要构建 `AST`，就需要利用到 有限状态机 的概念。

有限状态机也被叫做 **有限自动状态机**，表示：**有限个状态以及在这些状态之间的转移和动作等行为的数学计算模型**

光看概念，可能难以理解，那么下面我们来看一个具体的例子：

根据 `packages/compiler-core/src/compile.ts` 中的代码可知， `ast` 对象的生成是通过 `baseParse` 方法得到的。

而对于 `baseParse` 方法而言，接收一个 `template` 作为参数，返回一个 `ast` 对象。

即：**通过 `parse` 方法，解析 `template`，得到 `ast` 对象。** 中间解析的过程，就需要使用到 **有限自动状态机。**

我们来如下模板 （`template`） ：

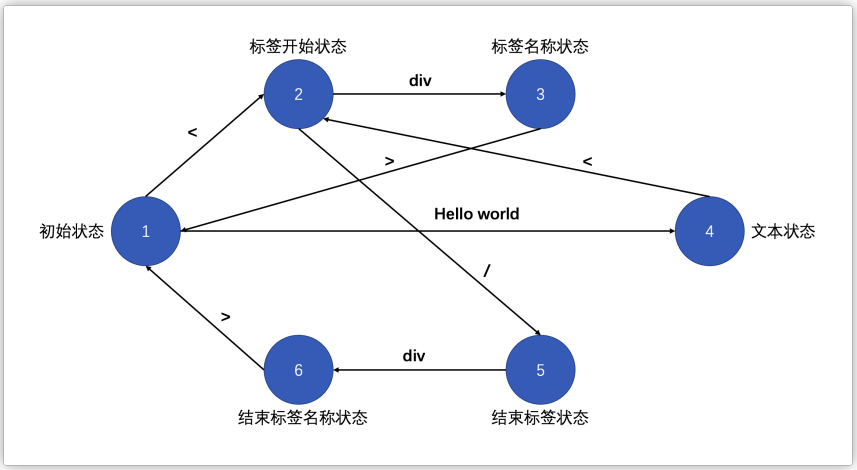
代码块

```
1    <div>hello world</div>
```

`vue` 想要把该模板解析成 `AST`，那么就需要利用有限自动状态机对该模板进行分析，分析的过程中主要包含了三个特性：

> 摘自：http://www.ruanyifeng.com/blog/2013/09/finite-state_machine_for_javascript.html
>
> 1. 状态总数是有限的
>
>   1. 初始状态
>   2. 标签开始状态
>   3. 标签名称状态
>   4. 文本状态
>   5. 结束标签状态
>   6. 结束标签名称状态
>   7. …
>
> 2. 任一时刻，只处在一种状态之中
> 3. 某种条件下，会从一种状态转变到另一种状态
>
>   1. 比如：从 `1` 到 `2` 意味着从初始状态切换到了标签开始状态

如下图所示：



1. 解析 `<`：由 **初始状态** 进入 **标签开始状态**
2. 解析 `div`：由 **标签开始状态** 进入 **标签名称状态**
3. 解析 `>`：由 **标签名称状态** 进入 **初始状态**
4. 解析 `hello world`：由 **初始状态** 进入 **文本状态**
5. 解析 `<`：由 **文本状态** 进入 **标签开始状态**
6. 解析 `/`：由 **标签开始状态** 进入 **结束标签状态**
7. 解析 `div`：由 **结束标签状态** 进入 **结束标签名称状态**
8. 解析 `>`：由 **结束标签名称状态** 进入 **初始状态**

经过这样一些列的解析，对于：

代码块

```
<div>hello world</div>
```

而言，我们将得到三个 `token`：

代码块

```
开始标签：<div>
文本节点：hello world
结束标签：</div>
```

而这样一个利用有限自动状态机的状态迁移，来获取 `tokens` 的过程，可以叫做：**对模板的标记化。**

总结

那么这一小节，我们了解了什么是有限自动状态机，也知道了它的三个特性。

`vue` 利用它来实现了对模板的标记化，得到了对应的 `token`。

那么这些 `token` 有什么用呢？我们下一小节再说。

01：前言 上一节　　下一节 03：扩展知识：扫描 tokens 构建 AST 结构...

我要提出意见反馈