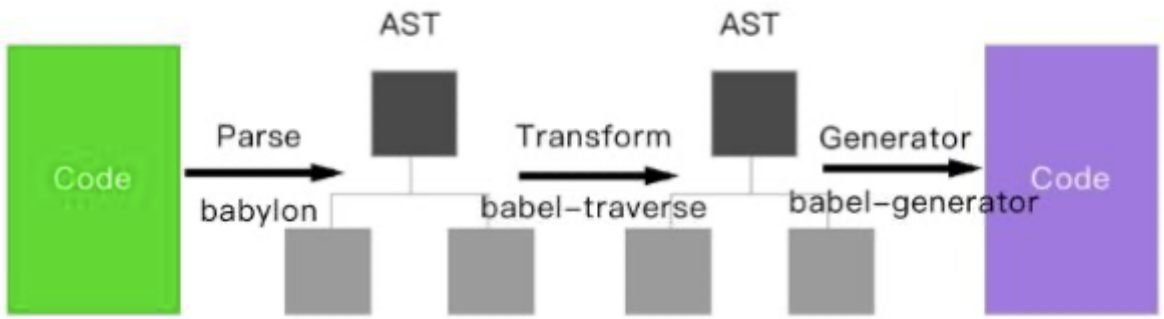
Babel 面试题

聊一聊 Babel 原理吧

大多数 JavaScript 遵循 estree 规范，Babel 最初基于 acorn 项目（轻量级现代 JavaScript 解析器），Babel 大概分为三大部分：

- 解析（Parse）：将代码转换成AST
  - 词法分析：将代码（字符串）分割成token流，即语法单元生成的数组
  - 语法分析：分析token流（上面生成的数组）并生成AST
- 转换（Transform）：对于 AST 进行变换一系列的操作，babel 接受得到AST 并通过 babel-traverse对其遍历，在此过程中进行添加、更新及移除等操作
  - Taro 就是利用 babel 完成的小程序语法转换
- 生成（Generate）：将变换后的 AST 再转换为 JS 代码，使用到的模块是 babel-generator



如何写一个 Babel 插件

Babel 解析成 AST，然后插件更改 AST，最后由 Babel 输出代码。那么 Babel 的插件模块需要你暴露一个 function，function 内返回 visitor

```
module.export = function (babel) {
    return {
        visitor: {}
    }
}
```

visitor 是对各类型的 AST 节点做处理的地方，那么我们怎么知道 Babel 生成的 AST 有哪些节点呢？很简单，你可以把 Babel 转换的结果打印出来，或者这里有传送门 AST explorer

Tree-shaking 原理

Tree-shaking 的本质是消除无用的js代码。无用代码消除在广泛存在于传统的编程语言编译器中，编译器可以判断出某些代码根本不影响输出，然后消除这些代码，这个称之为DCE（dead code elimination）

Tree-shaking 是 DCE 的一种新的实现，JavaScript 同传统的编程语言不同的是，JavaScript 绝大多数情况需要通过网络进行加载，然后执行，加载的文件大小越小，整体执行时间更短，所以去除无用代码以减少文件体积，对JavaScript来说更有意义

Tree-shaking 和传统的 DCE 的方法又不太一样，传统的 DCE 消灭不可能执行的代码，而 Tree-shaking 更关注消除没有用到的代码