

Let's make a compiler

Okay, first lets talk about compilers

- software that takes code written in one language and converts it to another
- PHP -> C
- C -> Machine Code

Why do we need compilers

- code written for humans is hard for computers to understand
- code written for computers is hard of humans to understand

Machine Code example:

```
11100101 11011001 11010010 11110001
```

Breaking a compiler down

- Front End - Takes source file and turns it into an intermediate representation
- Middle End - Makes performance optimisations
- Back End - Turns intermediate representation into the target code

Parts of the compiler we will look at

Front End

- Tokenizer (sometimes called a Lexer)
- Parser

Back End

- Code Generator

Tokenizer

Takes your code and turns it into understandable single chunks.

```
def hello()
```

becomes

```
token: def  
token: identifier (hello)  
token: open_parenthesis  
token: close_parenthesis
```

Parser

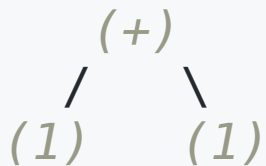
Turns tokens into an Intermediate Representation

1 + 1

becomes

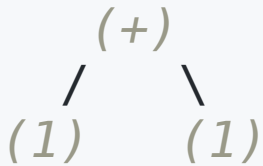
token: 1
token: +
token: 1

becomes



Code Generator

Takes our `Intermediate Representation` and turns it into the code we care about.



```
graph TD; A["(+ )"] -- "/" --> B["( 1 )"]; A -- "\" --> C["( 1 )"]
```

A diagram of an abstract syntax tree (AST) for the expression `(1) + (1)`. The root node is `(+)`, which has two children: `(1)` on the left and `(1)` on the right. The nodes are connected by a forward slash `/` and a backslash `\`.

becomes

```
add(1, 1)
```


Sidenote: Transpilers

- just another form of compiler. It keeps code at the same level of abstraction.
- Babel es2015 stuff
- what we're going to write today

Demo Time!

toby.pants << 

What we've covered

- a compiler is just a program, doing normal programmy things
- the main parts of a compiler
 - Front / Middle / Back ends
 - Tokenizer
 - Parser
 - Code Generator
- the difference between a Transpiler and a Compiler
- how we could make a (very quick and dirty) compiler

Questions / Thanks! / Simple CS

- thanks to Destroy All Software
- slides and code <https://github.com/tosbourn-ltd/compiler-talk>
- @tosbourn on twitter if you want to tweet questions later
- <https://tosbourn.com/simple-cs/>