Modules for C++

Daveed Vandevoorde

Modules for C++

C++

C++

C++ Libraries

Libraries

```
export MyLib:
public:
  typedef decltype(sizeof(1)) IndexType;
  ...
private:
  int select(IndexType x) {
    ...
  }
```

```
import MyLib;

void select(IndexType x) {
  ...
}
```

```
export M1:
public: typedef int I1;
```

```
export M2:
public: typedef int I2;
```

```
export MM:
public:  import M1;
private: import M2;
```

```
import MM;
I1 i1;  // Okay.
I2 i2;  // Error.  I2 is not declared.
```

```
export MyLib::Basics:
public:
  namespace MyLib {
    typedef decltype(sizeof(1)) IndexType;
  ...
```

```
import MyLib::Basics;

void select(MyLib::IndexType x) {
   ...
}
```

```
export MyLib.basics:
public:
  namespace MyLib {
    typedef decltype(sizeof(1)) IndexType;
private:
    void helper() {
  ...
```

```
export MyLib.select:
  import MyLib.basics;
public:
  namespace MyLib {
    void select(IndexType x) {
      ... helper(); ...
```

```
export MyLib:
public:
  struct S {
    void f(long x) { this->f((long long)x); }
  private:
    void f(long long x);
  };
  void S::f(long long x) { ... }
```

```
import MyLib;

void g(S x) {
  x.f(42);  // Unambiguous.
}
```

```
export MyLib:
public:
  class B {
    virtual void f();
  };
```

```
import MyLib;

struct D: B {
  override void f();  // Does find B::f!
};
```

```
export MyLib:
private:
  import {
    extern "C" int printf(char const*, ...);
  }
public:
  void warning(char const *msg) {
    printf("%s", msg);
  }
```

```
#ifndef MYLIB_H
#define MYLIB_H

import MyLib;

#endif  /* ifndef MYLIB_H */
```

So what?

Macros

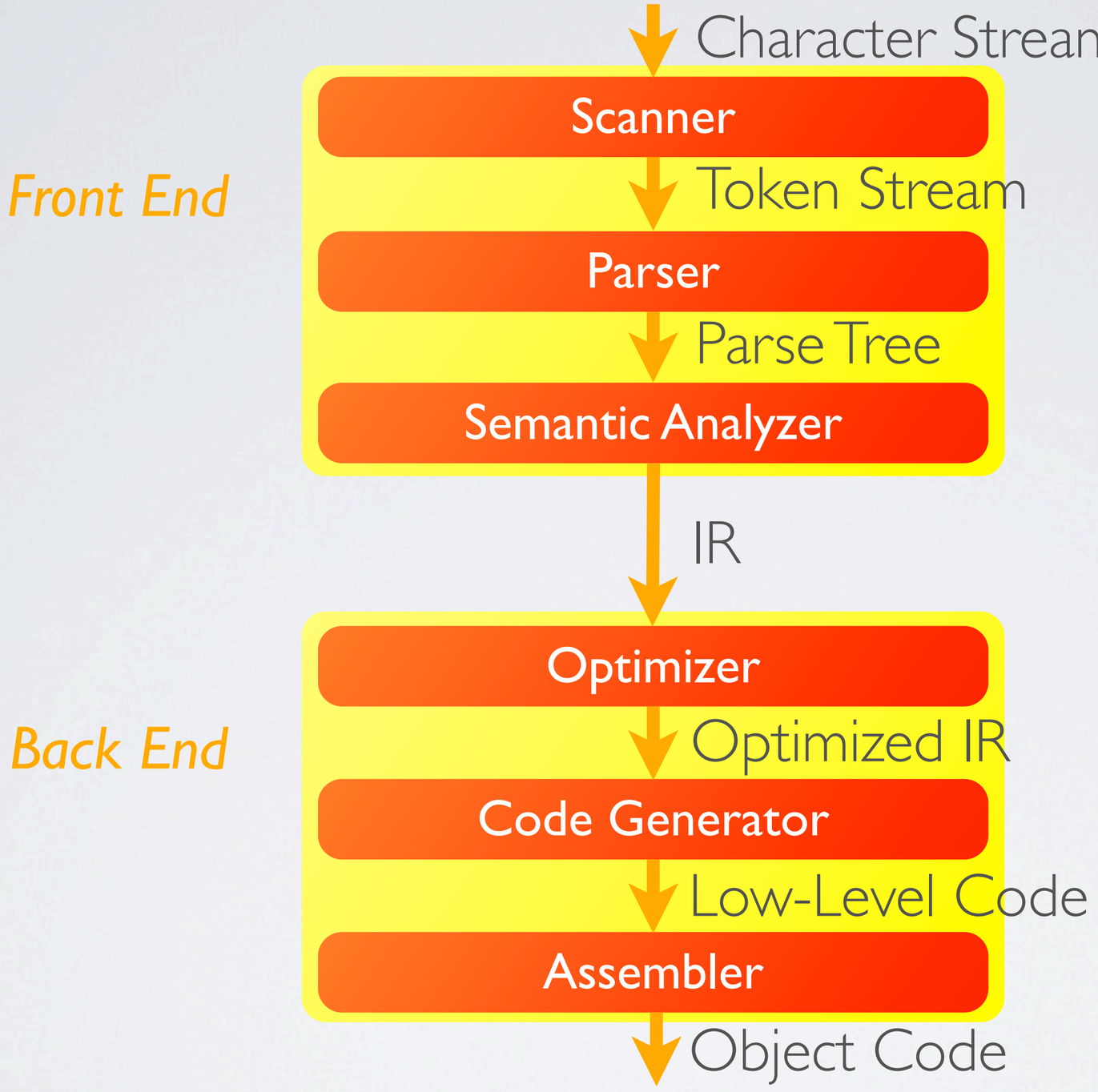

Macros

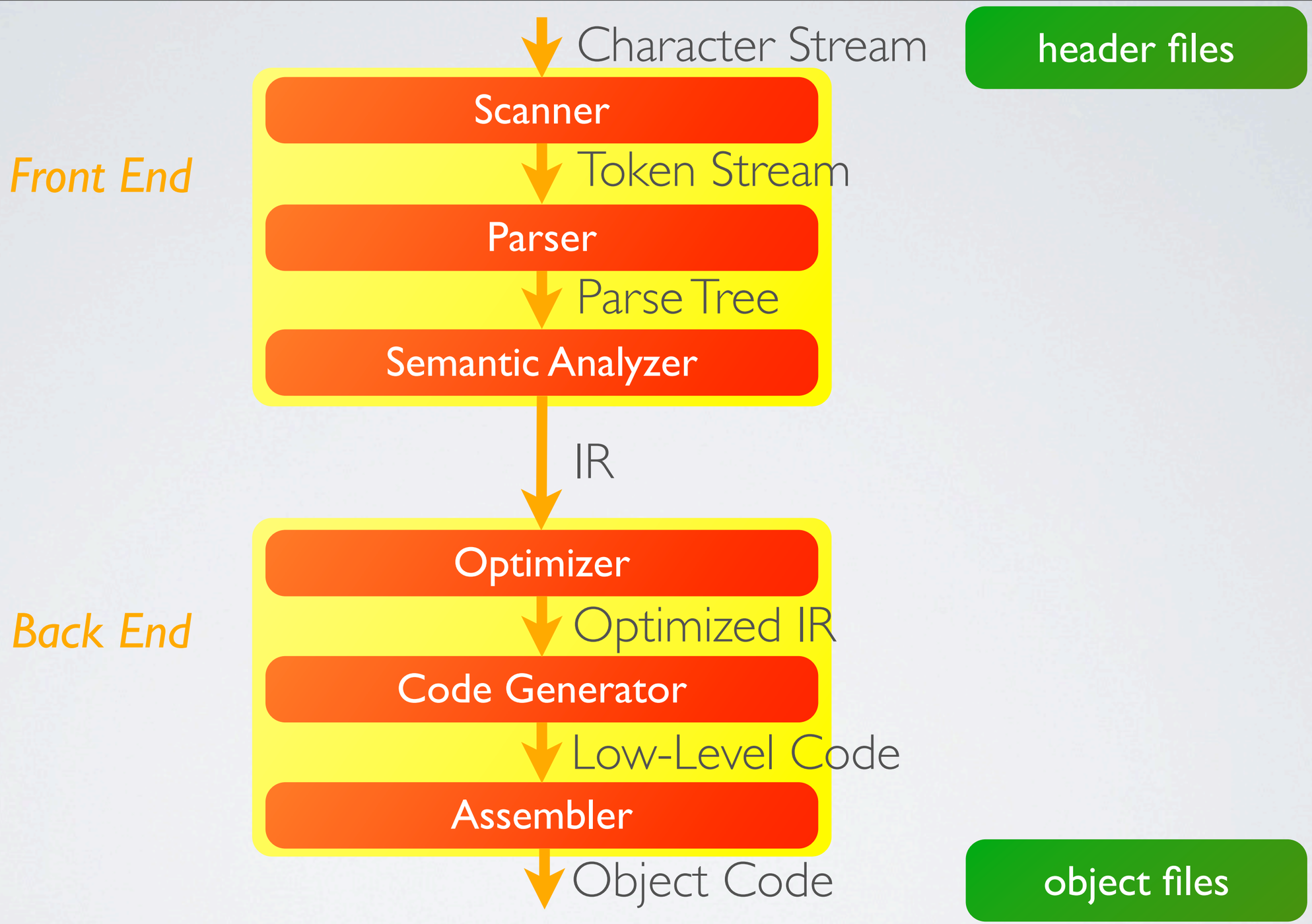
Fast Builds

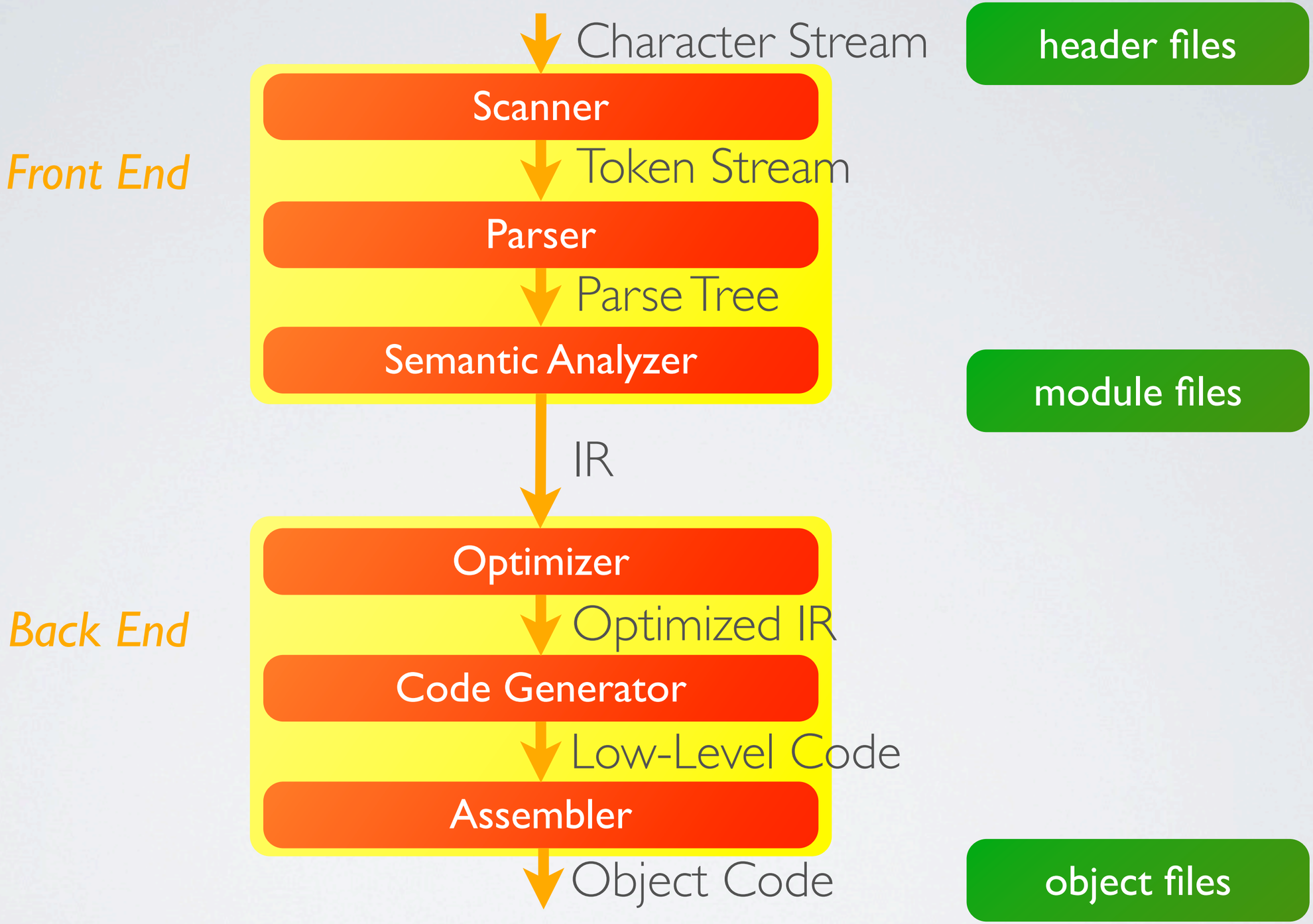
Explicit Dependencies

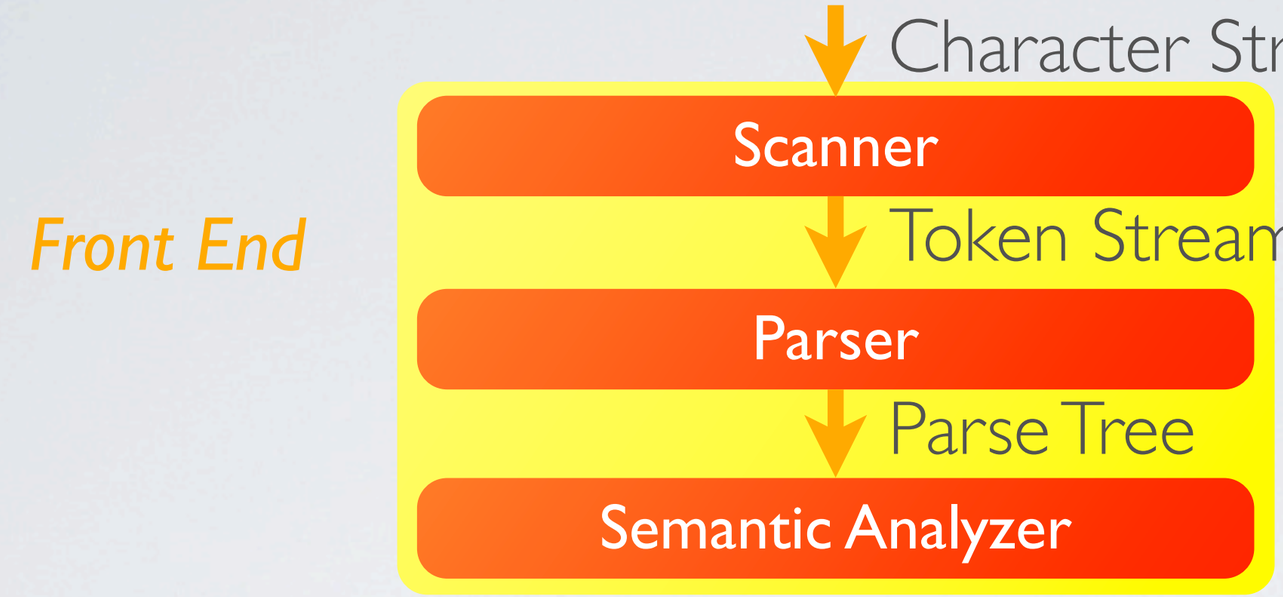
Initialization Order

Cross-TU Optimizations


Character Stream
Scanner
Front End
Token Stream
Parser
Parse Tree
Semantic Analyzer
IR
Optimizer
Back End
Optimized IR
Code Generator
Low-Level Code
Assembler
Object Code


Character Stream
header files
Front End
Scanner
Token Stream
Parser
Parse Tree
Semantic Analyzer
IR
Back End
Optimizer
Optimized IR
Code Generator
Low-Level Code
Assembler
Object Code
object files


Character Stream
header files
Front End
Scanner
Token Stream
Parser
Parse Tree
Semantic Analyzer
module files
IR
Optimizer
Optimized IR
Back End
Code Generator
Low-Level Code
Assembler
Object Code
object files

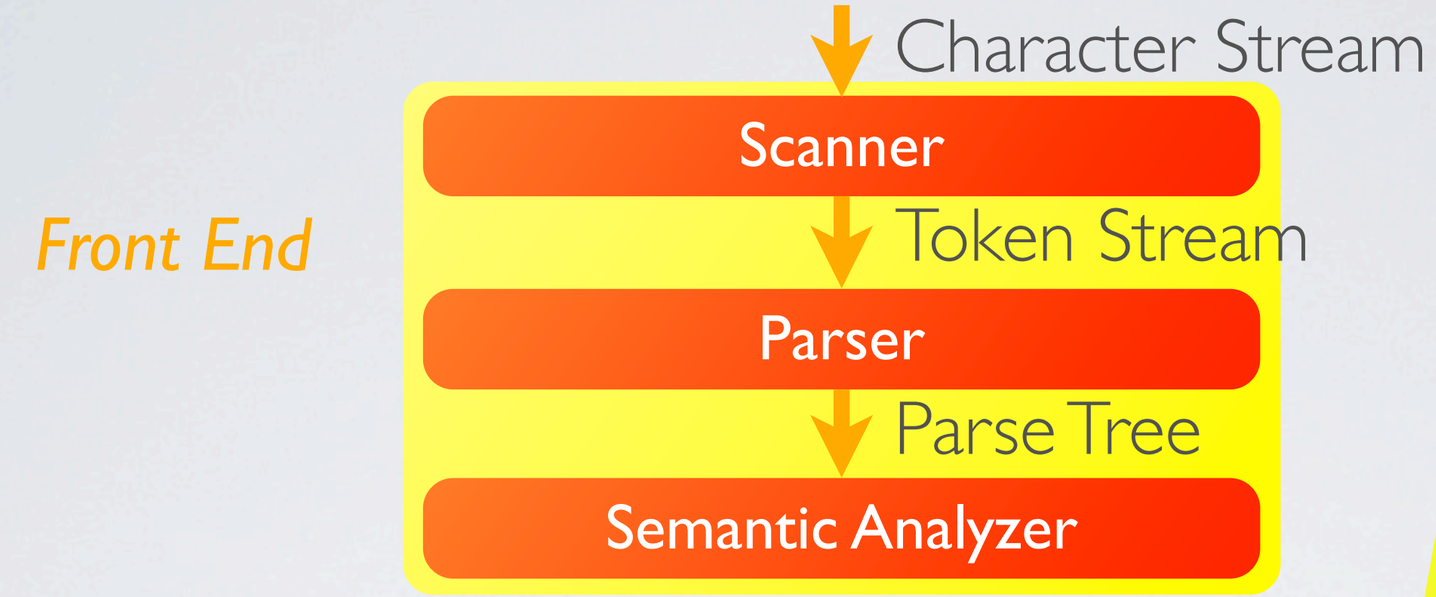


Front End

header files

module files

- *Compiler dependent*
- *Version dependent*
- *Human readable?*



Front End

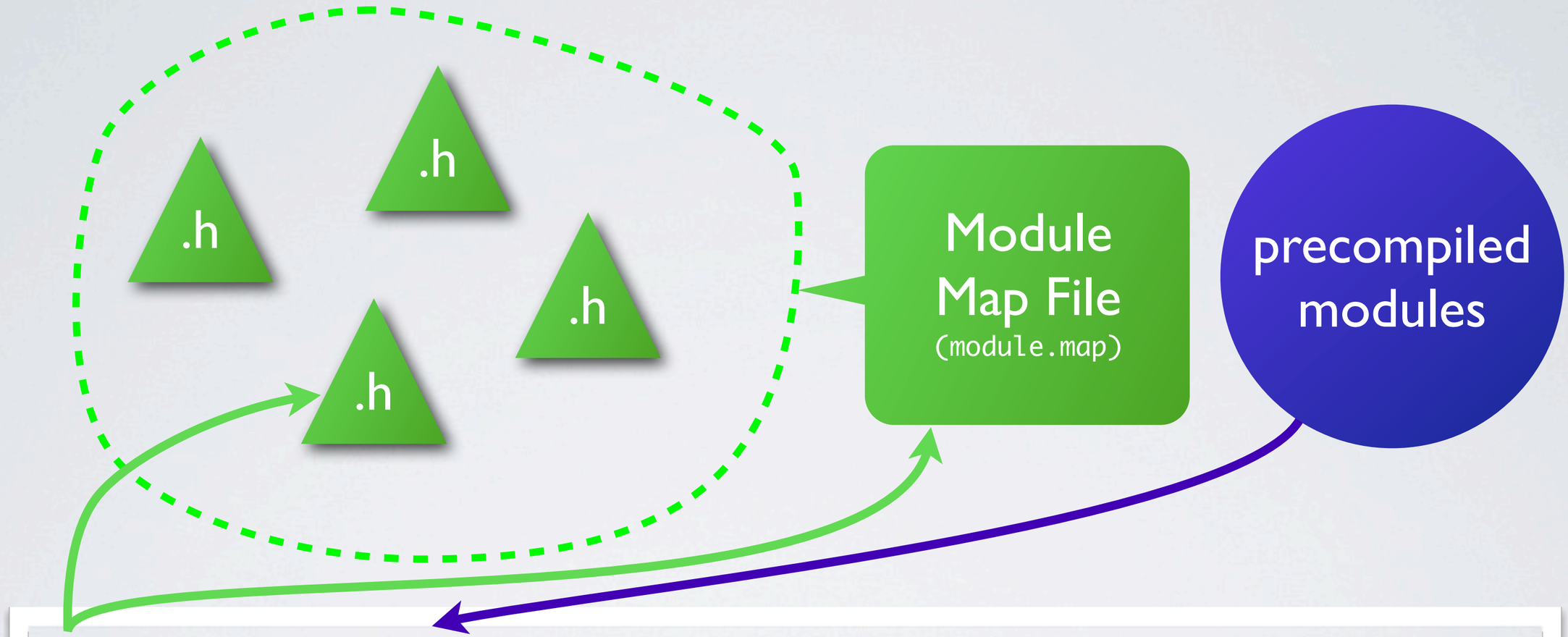
module files

precompiled modules

- *Compiler independent*
- *Version independent*
- *Human readable*

Modules

Clang Modules




```
#include “lib_part.h”

void select(MyLib::IndexType x) {
  ...
}
```

```
// module.map
module std {
  module stdio { header “stdio.h” }
  module vector {
    require cplusplus
    header “vector”
  }
  ...
  explicit module iostream {
    require cplusplus
    header “iostream”
    export std.locale
  }
  ...
}
```

Strengths & Weaknesses

Availability

Ease of Transition

Ease of Programming

Visibility Control

Initialization Ordering

Cross-TU Properties

Unifiable?

```
// module.map
module std {
  module stdio { header “stdio.h” }
  module vector {
    require cplusplus
    header “vector”
  }
  ...
  explicit module iostream {
    require cplusplus
    header “iostream”
    export std.locale
  }
  ...
}
```

```
// std.mpp
export [[mapincludes]] std:
  export [[macros]] .stdio {
#   include "stdio.h"
  }
  export [[cplusplus]] .vector {
#   include "vector"
  }
  ...
  export [[cplusplus, separate]] .iostream {
#   include "iostream"
  public:
    import std.locale;
  }
  ...
}
```

Want more?

SG-2

The END