Move Semantics: User-Defined Data Types

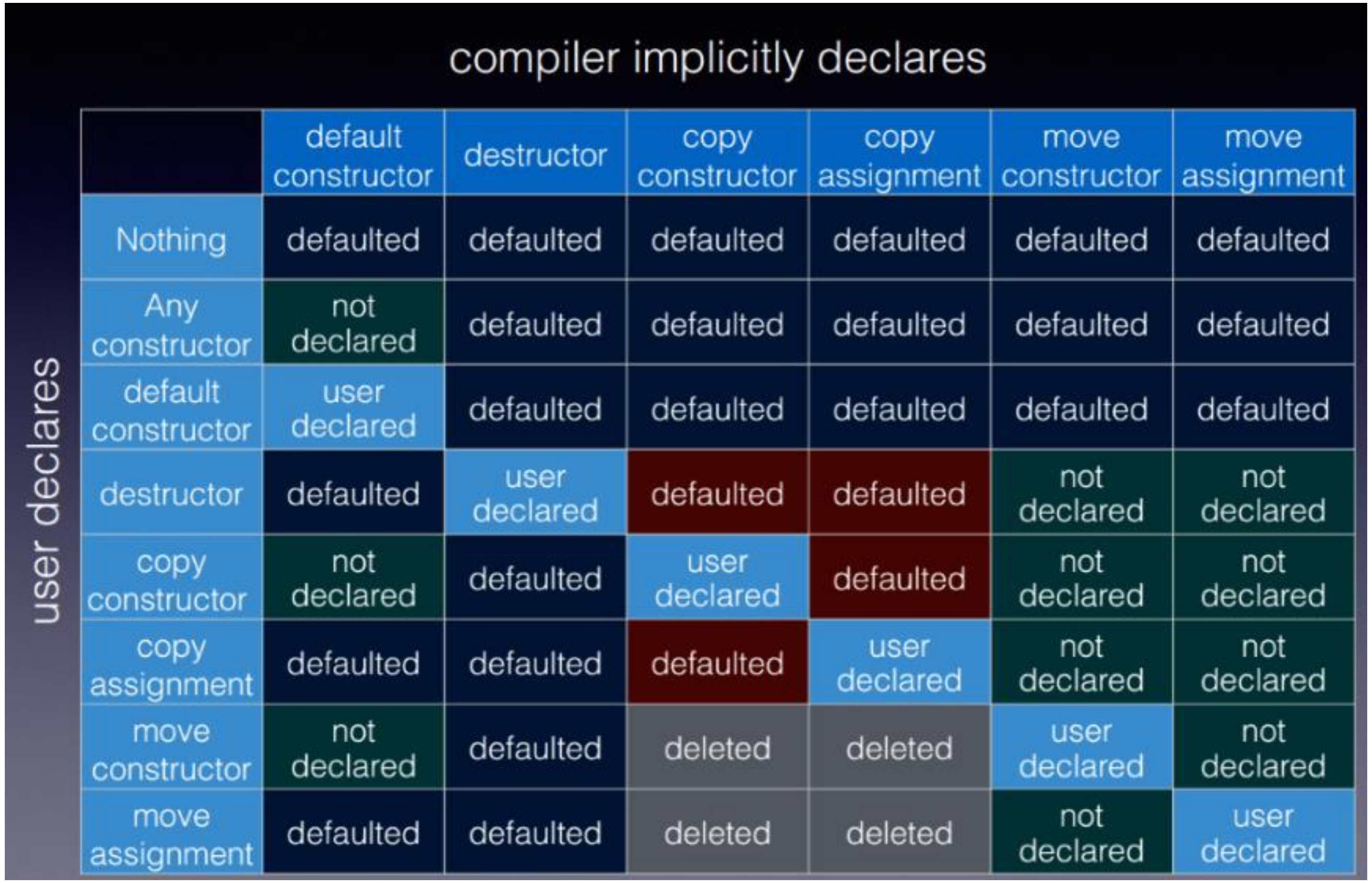
User defined data types can support move- and copy semantics.

- Example:

```
struct MyData{
    MyData() = default;
    MyData(MyData&& m) = default;
    MyData& operator = (MyData&& m) = default;
    MyData(const MyData& m) = default;
    MyData& operator = (const MyData& m) = default;
    ~MyData() = default;
};
```

  - Move semantics has higher priority than copy semantics.

Automatically Generated Member Functions



compiler implicitly declares

user declares

	default constructor	destructor	copy constructor	copy assignment	move constructor	move assignment
Nothing	defaulted	defaulted	defaulted	defaulted	defaulted	defaulted
Any constructor	not declared	defaulted	defaulted	defaulted	defaulted	defaulted
default constructor	user declared	defaulted	defaulted	defaulted	defaulted	defaulted
destructor	defaulted	user declared	defaulted	defaulted	not declared	not declared
copy constructor	not declared	defaulted	user declared	defaulted	not declared	not declared
copy assignment	defaulted	defaulted	defaulted	user declared	not declared	not declared
move constructor	not declared	defaulted	deleted	deleted	user declared	not declared
move assignment	defaulted	defaulted	deleted	deleted	not declared	user declared

by Howard Hinnant

- user-declared: a member function which is used (defined, `default`ed, or `delete`d)
- defaulted: a member function which the compiler generates or is requested via `default`