An import was unresolved.

Erroneous code example:

```
use something::Foo; // error: unresolved import `something::Foo`.
```

In Rust 2015, paths in `use` statements are relative to the crate root. To import items relative to the current and parent modules, use the `self::` and `super::` prefixes, respectively.

In Rust 2018, paths in `use` statements are relative to the current module unless they begin with the name of a crate or a literal `crate::`, in which case they start from the crate root. As in Rust 2015 code, the `self::` and `super::` prefixes refer to the current and parent modules respectively.

Also verify that you didn't misspell the import name and that the import exists in the module from where you tried to import it. Example:

```
use self::something::Foo; // Ok.

mod something {
    pub struct Foo;
}
# fn main() {}
```

If you tried to use a module from an external crate and are using Rust 2015, you may have missed the `extern crate` declaration (which is usually placed in the crate root):

```
extern crate core; // Required to use the `core` crate in Rust 2015.

use core::any;
# fn main() {}
```

In Rust 2018 the `extern crate` declaration is not required and you can instead just `use` it:

```
use core::any; // No extern crate required in Rust 2018.
# fn main() {}
```