use of its individual enumerators, and (2) typical components consist of just a `.h`/`.cpp` pair, i.e., exactly one `.h` file and usually just one `.cpp` file.[10]

Inciting local enumeration declarations: An attractive nuisance

Whenever we, as library component authors, provide the complete definition of an enumeration with a fixed underlying type and fail to provide a corresponding forwarding header having just the opaque declaration, we confront our clients with the difficult decision of whether to needlessly compile-time couple[11] themselves and/or their clients with the details of the enumerator list or to make the dubious choice to unilaterally redeclare that enumeration locally.

The problems associated with local declarations of data whose types are maintained in separate translation units is not limited to enumerations; see *Redeclaring an externally defined enumeration locally* on page 675. The maintainability pitfall associated with opaque enumerations, however, is qualitatively more severe than for other external-linkage types, such as a global **`int`**, in that the ability to elide the enumerators amounts to an *attractive nuisance* wherein a client — wanting to do so and having access to only a single header containing the complete definition — might be persuaded into declaring the enumeration locally!

Ensuring that library components that define enumerations whose enumerators can be elided also consistently provide a second forwarding header file containing the opaque declaration of each such enumeration would be one generally applicable way to sidestep this maintenance burden; see *Use Cases — Dual-Access: Insulating some external clients from the enumerator list* on page 665. Note that the attractive nuisance potentially exists even when the primary intent of the component is not to make the enumeration generally available.[12]

Annoyances

Opaque enumerations are not completely type safe

Making an enumeration opaque does not stop it from being used to create an object that is initialized opaquely to a zero value and then subsequently used (e.g., in a function call):

```
enum Bozo : int;  // forward declaration of enumeration Bozo
void f(Bozo);     // forward declaration of function f

void g()
{
```

[10] **lakos20**, sections 2.2.11–2.2.13, pp. 280–281

[11] For a complete real-world example of how compile-time coupling can delay a "hot fix" by weeks, not just hours, see **lakos20**, section 3.10.5, pp. 783–789.

[12] **wight**