**Anonymous unions.**

When *unions* are members of a *class* (or *structure*), they can be declared with no name. In this case, they become *anonymous unions*, and its members are directly accessible from objects by their member names. For example, see the differences between these two structure declarations:

|  |  |
| --- | --- |
| **structure with regular union** | **structure with anonymous union** |
| struct book1\_t {  char title[50];  char author[50];  union {  float dollars;  int yen;  } price;  } book1; | struct book2\_t {  char title[50];  char author[50];  union {  float dollars;  int yen;  };  } book2; |

The only difference between the two types is that in the first one, the member union has a name (*price*), while in the second it has not. This affects the way to access members dollars and yen of an object of this type. For an object of the first type (with a regular *union*), it would be:

|  |  |  |
| --- | --- | --- |
| 1 2 | book1.price.dollars  book1.price.yen |  |

whereas for an object of the second type (which has an anonymous *union*), it would be:

|  |  |  |
| --- | --- | --- |
| 1 2 | book2.dollars  book2.yen |  |

Again, remember that because it is a *member union* (not a *member structure*), the members dollars and yen actually share the same memory location, so they cannot be used to store two different values simultaneously. The price can be set in dollars or in yen, but not in both simultaneously.