Template Instantiation

Instantiating a template means using the template to create a function using specific types for its template parameters. There are two ways to do this, **implicitly** and **explicitly**. The functions generated by a pararticular instantiation are called **template functions** (which is, unfortunately, easily confused with function template).

Let's look at another example:

```
template <typename T>
void print(const T& val)
{
    cout << val;
}
```

When you **call the function**, the compiler will **implicitly deduce** the types of arguments you pass and then generate and call a **version of the function** with those parameters.

```
string s = "hello";
print(23);        // generates print(const int&)
print(str);       // generates print(const string&)
```

The first call **implicitly** instantiates (and calls) a **`print(int)`** function, while the second generates and calls a **`print(string)`** template function.