# Material para templates

## Notas importantes

1. *Visual Studio* odia las **templates**, así que es bastante probable que quiera fallar porque sí. En esos casos me preguntan, ya que tengo experiencia muriendo con eso.
2. Si uno utiliza templates en su clase, la clase no puede ser dividida en `.cpp` y `.h`, todo se debe hacer en el archivo `.h`.
3. Si se usa **templates** y se desea separar la declaración de la implementación (i.e. poner dentro de la clase `int funcion();` y luego de la clase poner el la implementación del método) se debe poner `Clase<T>::metodo()` en lugar de solo `Clase::metodo()`.

## FAQ

### ¿Para qué sirven las templates?

Las templates en su mayoría sirven para hacer estructuras de datos genéricas, que funcionen para cualquier tipo de dato, ya sea creado por el usuario o primitivo.

### ¿Cómo implemento una clase con template?

Este es un ejemplo que pueden agarrar como base:

```
template <class T>
class clase {
    private:
        T atributo;
    public:
        clase(T);
        T getAtributo();
        void setAtributo(T);
};

clase<T>::clase(T atributo) {
    this->atributo = atributo;
}

T clase<T>::getAtributo() {
    return this->atributo;
}

void clase<T>::setAtributo(T atributo) {
    this->atributo = atributo;
}
```

### ¿Cómo uso mi clase template una vez la declaré?

Usando como base el ejemplo anterior:

```
int main() {
    clase<int> objeto(22);
    std::cout << objeto.getAtributo() << std::endl;
    // Aquí se imprime 22
    objeto.setAtributo(99);
    std::cout << objeto.getAtributo() << std::endl;
    // Aquí se imprime 99.
}
```

## ¿Puedo usar algo diferente a T para el tipo?

Sí, no hay problema con el nombre del tipo a usar.

Pero es mejor usar un nombre corto, para que no les pase lo del segundo ejemplo.

**Primer ejemplo:**

```
template <class T>
class clase {
    private:
        T atributo;
    public:
        clase(T);
};
```

**Segundo ejemplo:**

```
template <class tipo_con_nombre_raro>
class clase_con_template_con_tipo_raro {
    private:
        tipo_con_nombre_raro atributo;
    public:
        clase_con_template_con_tipo_raro(tipo_con_nombre_raro);
};
```

## ¿Cuál es la diferencia entre class y typename usando templates?

Ninguna, uno puede usarlos como le plazca, el más bonito.

**Ejemplo:**

```
template <class identifier> function_declaration;
template <typename identifier> function_declaration;
```

Cualquiera de los dos funciona igual.

### ¿Puedo usar templates con una función?

Claro, un ejemplo de esto es el siguiente:

```
template <class T>
T GetMax (T a, T b) {
    return (a > b ? a : b);
}
```

### ¿Puedo usar una template con más de un tipo?

Sí, esto es usado en su mayoría para hacer estructuras como *Tuplas*, *Pares* y *Mapas*.

**Ejemplo:**

```
template <class T1, class T2>
struct Tupla {
    public:
        T1 atributo1;
        T2 atributo2;
};
```

## Material de refenrecia:

- http://www.cplusplus.com/doc/oldtutorial/templates/.