Sync in C++: *Sharable between threads*

92

```cpp
template<typename T>
struct is_sync : std::false_type {};

template<typename T>
struct is_sync<std::atomic<T>> : std::true_type {};

template<typename T>
inline constexpr bool is_sync_v = is_sync<T>::value;

template<typename... Args>
concept sync = (is_sync<Args>::value && ...);
```

```
static_assert(! is_sync_v<int>);
static_assert(! is_sync_v<int&>);
static_assert(! is_sync_v<const int&>);
static_assert(! is_sync_v<std::string&>);
static_assert(! is_sync_v<const std::string&>);
static_assert(is_sync_v<std::atomic<int>>);
```

Sync in C++: *Sharable between threads*

```cpp
template<typename T>
struct is_sync : std::false_type {};

template<typename T>
struct is_sync<std::atomic<T>> : std::true_type {};

template<typename T>
inline constexpr bool is_sync_v = is_sync<T>::value;

template<typename... Args>
concept sync = (is_sync<Args>::value && ...);
```

```cpp
static_assert(! is_sync_v<int>);
static_assert(! is_sync_v<int&>);
static_assert(! is_sync_v<const int&>);
static_assert(! is_sync_v<std::string&>);
static_assert(! is_sync_v<const std::string&>);
static_assert(is_sync_v<std::atomic<int>>);
```