dhorn service\_container

# Introduction

The dhorn::service\_container class is meant to provide an easy way of publishing and looking up needed services. The service container stores one instance per type.

# Classes/Structures/Types/Enumerations/etc.

The service container is split between the main dhorn::service\_container class, and different traits structs.

## dhorn::service\_conatiner class

The service container class essentially consists of three classes of functions: inserting, retrieving, and removing. Inserting can be done either by passing in a pointer to the object, or by emplacing a new object.

class service\_container final  
{

public:  
 service\_container(void);

service\_container(\_In\_ const service\_container &) = delete;

service\_container(\_Inout\_ service\_container &&);

/\*unspecified\*/ operator=(\_In\_ const service\_container &) = delete;

service\_container &operator=(\_Inout\_ service\_container &&);

~service\_container(void);

template <typename Ty>

Ty \*find(void);

template <typename Ty, template Deleter = std::default\_delete<Ty>>

void insert(\_In\_ Ty \*p, \_In\_ const Deleter &deleter = Deleter());

template <typename Ty, template Deleter = std::default\_delete<Ty>>

void insert(\_In\_ Ty \*p, \_Inout\_ Deleter &&deleter = Deleter());  
}