BasicPortAllocator Class

A subclass of the **PortAllocator** class, **BasicPortAllocator** handles port allocation. You would modify or override this class if you introduce new types of ports, or want more control of which candidates are allocated, which network cards, or other details. The methods shown are not thread-safe, and can be called on any thread.

This class is extended by **HttpPortAllocator**, which handles negotiation of stun and relay hosts. You should use **HttpPortAllocator** unless you have special needs or have extended **BasicPortAllocator** yourself.

Syntax

```
class BasicPortAllocator: public PortAllocator
```

Name	Description
void **AddWritablePhase**(int *phase*)	Called whenever a connection becomes writable, with *phase* being the phase that the corresponding port was created in.
BasicPortAllocator(NetworkManager **network_manager*)	Simple constructor. Pass in the **NetworkManager** object created by the application.
BasicPortAllocator(NetworkManager **network_manager*, SocketAddress **stun_server*, SocketAddress **relay_server*)	Constructor with optional STUN and relay server addresses. The example code uses the Google Talk addresses, but you should use different addresses in your code. • *network_manager* The **NetworkManager** object created by the application. • *stun_server* Optional address of a STUN server to use to request your NAT-enabled device's external address. In production code, if not using the Google Talk server you should build and run your own STUN server from the **cricket::StunServer** class. • *relay_server* Optional address of a relay server to use for the data channel. libjingle provides code (relayserver.h/cc) to build your own relay server. The example code does not use a relay server.

int **best_writable_phase**()	Returns the best (highest preference) phase that has produced a port that produced a writable connection. If no writable connections have been produced, this returns -1.
	
	The following phases are defined in basicportallocator.cc:
	
	const int PHASE_UDP = 0;
	
	const int PHASE_RELAY = 1;
	
	const int PHASE_TCP = 2;
	
	const int PHASE_SSLTCP = 3;

PortAllocatorSession* **CreateSession**(const std::string &*name*, const std::string& *session_type*)	Creates a helper object that handles much of the port allocation. *session_type* must be a unique value for that specific session type, and must match the value specified in the *session_type* parameter of **SessionManager::AddClient** for that session type.
NetworkManager* **network_manager**()	Returns the **NetworkManager** object used to instantiate this object.

Attributes: public

Declaration file: talk/p2p/client/basicportallocator.h