Wt - Feature #5307

Possibility of setting a different thread_pool count for main process and session processes, or detect if I'm a session process or not

09/29/2016 04:55 PM - Aarón Bueno

Status:	New	**Start date:**	09/29/2016
Priority:	Normal	**Due date:**	
Assignee:		**% Done:**	0%
Category:		**Estimated time:**	0.00 hour
Target version:			

Description

(Sorry for my English) At least when using the wthttpd connector in dedicated process mode, the configured thread-pool size (wt_config.xml, wthttpd config file or command line parameter) is the same for the main and any of the session processes.

However, the main process is outside of user control, but any session process is where user code lives, and perhaps the user wants to customize its own thread-pool, for any reason (for example, to shrink the thread-pool to reduce parallelism, if he is unsure on synchronization issues). Of course, you can set programatically the thread-pool size through `server.ioService().setThreadCount(your_size)`, but that sets the `thread-pool` for main and session servers, again.

My "trick" to config the `thread count` separately was to check who am I, whether a main process or server process, looking for the `---parent-port` parameter:

```
Wt::WServer server(argc, argv, WTHTTP_CONFIGURATION);

std::string arg(argv[argc - 1]);

auto* test = "---parent-port";

// If I'm a session process

if (arg.compare(0, strlen(test), test)) {

std::string thread_count;

server.readConfigurationProperty("session-process-thread-count", thread_count); // I added it as  in my wt_config.xml

server.ioService().setThreadCount(std::stoi(thread_count));

}
```

My feature request can be any of:

*) Adding an extra configuration option to wt_config.xml, for example: `dispatcher-thread-pool`, as suggested by Koen Deforche by email. Only if present, it's used as thread-pool size of session processes.

*) Alternately, or in addition, a set of functions to know if I'm a main or session process, connector type, and so on, to allow the user makes any kind of customizations according to general server/process properties. I would do it through a WServer member function returning an object of a `WServerInfo` class or something like that. In the case of the wthttpd connector, it could be implemented through a wrapper class of http::server::Configuration and/or Wt::Configuration with only getters.

```
// connector_type is a enum of possible connectors: httpd, fcgi, isapi

ConnectorType WServer::connector_type() const;

const WServerInfo& WServer::serverInfo() const;
```

Initially, that Wt::WServerInfo class can have just two or three function for very basic server information: whether dedicated/shared mode, if "I'm" a main or child server, and so on. If more info is needed, the `WServer` remains untouched; you only need to change the `WServerInfo` interface.