Extend Server Configuration

English (United States) v1.6

Keywords: Configuration, Custom Configuration, Extend Configuration

When you implement your socket server by SuperSocket, it is unavoidable to define some parameters in configuration file.The SuperSocket provides a very easy way to store the parameters in your configuration file and then read and use them in AppServer.

Please take a look at the following configuration code:

```
<server name="FlashPolicyServer"
        serverType="SuperSocket.Facility.PolicyServer.FlashPolicyServer, SuperSocket.Facility"
        ip="Any" port="843"
        receiveBufferSize="32"
        maxConnectionNumber="100"
        clearIdleSession="true"
        policyFile="Policy\flash.xml">
</server>
```

In above server configuration,the attribute "policyFile" is not defined in SuperSocket, but you also can read it in your AppServer class:

```
public class YourAppServer : AppServer
{
    private string m_PolicyFile;

    protected override bool Setup(IRootConfig rootConfig, IServerConfig config)
    {
        m_PolicyFile = config.Options.GetValue("policyFile");

        if (string.IsNullOrEmpty(m_PolicyFile))
        {
            if(Logger.IsErrorEnabled)
                Logger.Error("Configuration option policyFile is required!");
            return false;
        }

        return true;
    }
}
```

Not only we can add customized attributes in server node, we also can add the customized child configuration like below:

```
<server name="SuperWebSocket"
        serverTypeName="SuperWebSocket"
        ip="Any" port="2011" mode="Tcp">
    <subProtocols>
        <!--Your configuration-->
    </subProtocols>
</server>
```

A configuration element type is required:

```
/// <summary>
/// SubProtocol configuration
/// </summary>
public class SubProtocolConfig : ConfigurationElement
{
    //Configuration attributes
}
/// <summary>
/// SubProtocol configuation collection
/// </summary>
[ConfigurationCollection(typeof(SubProtocolConfig))]
public class SubProtocolConfigCollection : ConfigurationElementCollection
{
    //Configuration attributes
}
```

Then you can read the child configuration node in your AppServer:

```
public class YourAppServer : AppServer
{
    private SubProtocolConfigCollection m_SubProtocols;

    protected override bool Setup(IRootConfig rootConfig, IServerConfig config)
    {
        m_SubProtocols = config.GetChildConfig<SubProtocolConfigCollection>("subProtocols");

        if (m_SubProtocols == null)
        {
            if(Logger.IsErrorEnabled)
                Logger.Error("The child configuration node 'subProtocols' is required!");
            return false;
        }

        return true;
    }
}
```


- Prev: Push Data to Clients from Server Initiative (/v1-6/en-US/Push-Data-to-Clients-from-Server-Initiatively)
- Next: Server Configuration Hot Update (/v1-6/en-US/Server-Configuration-Hot-Update)