Cardinal-Anticheat Addons

Addons

1. First things first, download the `Extension.jar` from the dependencies
2. Add `Extension.jar` as your dependency. It's **only** a placeholder and does **not** belong inside your `plugins` folder
3. Instead of `extends JavaPlugin` you have to `implement Addon` which requires two methods:

```
import me.clientastisch.extension.Extension;
import me.clientastisch.extension.impl.Addon;

public class Core implements Addon {

    @Override
    public void onEnable() throws Exception {

    }

    @Override
    public void onDisable() throws Exception {

    }
}
```

4. Unlike spigot, a file called `addon.json` must be created like following:

```
{
  "name": "MyAwesomeAddon",
  "author": "Me",
  "main": "my.path.to.Core",
  "version": "v1.0"
}
```

5. Export your addon to `plugins//CAC//addons`
6. Reload your server or use `/cac addon load <addon>` and you're done!

Events

As you already know, you have to implement `Listener` in a class you want to use BukkitEvents. In case you also want to use the build-in events of the Cardinal-Anticheat you have to implement `EventListener`. That's because Cardinal-Anticheat has an entire different event-system. It's possible to use BukkitEvents and CardinalEvents in the same class by implementing both classes. However, I do not recomment that, due to the performance lose on startup.

```
import me.clientastisch.extension.impl.event.EventListener;
import me.clientastisch.events.EventManager;

public class MyWonderfulEvent implements EventListener {

}
```

The next difference is the `@EventHandler` which isn't used for CardinalEvents. Cardinal-Anticheat uses the annotation `@EventManager.Target` above methods:

```
@Retention(RetentionPolicy.RUNTIME)
@java.lang.annotation.Target(ElementType.METHOD)
public static @interface Target {

    boolean isSmart() default true;

    boolean isSync() default false;

    int sleep() default 0;
}
```

As you may have already noticed, there're some additional options which you don't have with the BukkitEvents. First things first, if `isSmart` is enabled methods which have already throwen an expcetion won't be invoked anymore. `isSync` invokes the method either asynchronous or synchronous to the event call. In case `isSync = false` you have the possibility to call the method in a delay by changing `delay`.

```
@EventManager.Target(isSync = true)
public void onFlag(CheckFireEvent event) {
    event.cancelled();
}
```

Keep in mind, that asynchronous events **cannot** be `cancelled`. Therefore set `isSync` to `true`.

Commands

To handle commands you have to implement `Command`

```
public interface Command {

    boolean execute(CommandSender sender, String command, String[] arguments, String raw);

}
```

In case your command matches `return true`

Register events and commands

You can register events and commands by accessing `Extension`.

```
import me.clientastisch.extension.Extension;
import me.clientastisch.extension.impl.Addon;

public class Core implements Addon {

    @Override
    public void onEnable() throws Exception {
        Extension.registerListener(this, new MyWonderfulEvent());
        Extension.registerCommand(this, new MyWonderfulCommand());
    }

    @Override
    public void onDisable() throws Exception {

    }
}
```

Why use addons

Addons are supported for every Spigot version on which Cardinal-Anticheat is supported too. This gives you the ability to create multiversion extensions. Futhermore, you get access to a bunch of events which are either packet or spigot based. You also get access to some player-data collected by the anticheat which gives you some additional information which spigot doesn't provide on its own.