IO Monad - ZIO

Sample environment module

```
trait HttpClient {
  val httpClient : HttpClient.Service[Any with HttpClient with Blocking]

}
object HttpClient {

  trait Service[R <: HttpClient with Blocking] {
    def get(url: String): ZIO[R, Throwable, String]
  }
```

IO Monad - ZIO

Test module

```scala
// The test http runtime
trait HttpClientTest extends HttpClient {

  val sampleTopStories = "[19536375,19536173,19535059,19535564]"
  val sampleItem = """{"by":"justinhj","id":11498534,"parent":11498393,"text":"I&#x27;m an emacs user but I often use a (

  val httpClient: Service[Any with HttpClient with Blocking] = new Service[Any with HttpClient with Blocking] {

    def requestSync(url: String) : String = {
      if(url == HNApi.getTopItemsURL) sampleTopStories
      else if(url == HNApi.getItemURL( itemId = 11498534)) sampleItem
      else throw new Exception(s"$url not found in http mock client")
    }

    final def get(url: String) : Task[String] = {
      ZIO.effect(requestSync(url))
    }
  }
}

object HttpClientTest extends HttpClientTest

trait TestRuntime extends Runtime[Clock with Console with System with Random with Blocking with HttpClient] {
  type Environment = Clock with Console with System with Random with Blocking with HttpClient

  val Platform: Platform       = PlatformLive.Default
  val Environment: Environment = new Clock.Live with Console.Live with System.Live with Random.Live with Blocking.Live
    with HttpClientTest
}
```