Working on per partition basis

Function name	We are called with	We return	Function signature on RDD[T]
mapPartitions()	Iterator of the elements in that partition	Iterator of our return elements	f: (Iterator[T]) → Iterator[U]

```
import org.eclipse.jetty.client.ContentExchange
import org.eclipse.jetty.client.HttpClient

object BasicMapPartitions {
    def main(args: Array[String]) {
      val master = args.length match {
        case x: Int if x > 0 => args(0)
        case _ => "local"
      }
      val sc = new SparkContext(master, "BasicMapPartitions", System.getenv("SPARK_HOME"))
      val input = sc.parallelize(List("KK6JKQ", "Ve3UoW", "kk6jlk", "W6BB"))
      val result = input.mapPartitions{
        signs =>
        val client = new HttpClient()
        client.start()
        signs.map {sign =>
          val exchange = new ContentExchange(true);
          exchange.setURL(s"http://qrzcq.com/call/${sign}")
          client.send(exchange)
          exchange
        }.map{ exchange =>
          exchange.waitForDone();
          exchange.getResponseContent()
        }
      }
      println(result.collect().mkString(","))
    }
}
```