**Before starting work , run following comands**

1.composer dump

2.composer install

3.composer update

**Class CheckAmazoneInstances**

**Private properties:**

**@var** aray private $config = []; Keeps the main configuration data

**@var** array private $runnedInstances = []; Stores the number of main region instances for scanning

**public function \_\_construct ($config)**

1. Initialaizing private configuration properties

**@param *array*** $config - contains array with configuration data

**@return** void

**public function run()**

This is a main action

Iterating by regions ,call private method getRunnedInstances .

Save json file that is specified in the main configuration file,

and before every calling getRunnedInstances method took out the stored data from json file,

comparing them instanceId,and if the time exceeds the permissible call the private method notify.

**@return** void

**private function getRunnedInstances($region)**

Get runned Instances by region

**@param** $region

**@return** arrayt Runned instances

**private function notify($notices)**

Generate a message based on an array of $notices to send email and logging

Send mail with notice mssage

Writ notice mssage into monitorLog.txt file

**@param** array $notices

**@return** void

**class Logger**

**github link** <https://github.com/Seldaek/monolog/tree/master/src/Monolog>

Monolog sends your logs to files, sockets, inboxes, databases and various web services. See the complete list of handlers below. Special handlers allow you to build advanced logging strategies.

This library implements the PSR-3 interface that you can type-hint against in your own libraries to keep a maximum of interoperability. You can also use it in your applications to make sure you can always use another compatible logger at a later time. As of 1.11.0 Monolog public APIs will also accept PSR-3 log levels. Internally Monolog still uses its own level scheme since it predates PSR-3.