<?php

/\*\*

\* EXHIBIT A. Common Public Attribution License Version 1.0

\* The contents of this file are subject to the Common Public Attribution License Version 1.0 (the “License”);

\* you may not use this file except in compliance with the License. You may obtain a copy of the License at

\* http://www.oxwall.org/license. The License is based on the Mozilla Public License Version 1.1

\* but Sections 14 and 15 have been added to cover use of software over a computer network and provide for

\* limited attribution for the Original Developer. In addition, Exhibit A has been modified to be consistent

\* with Exhibit B. Software distributed under the License is distributed on an “AS IS” basis,

\* WITHOUT WARRANTY OF ANY KIND, either express or implied. See the License for the specific language

\* governing rights and limitations under the License. The Original Code is Oxwall software.

\* The Initial Developer of the Original Code is Oxwall Foundation (http://www.oxwall.org/foundation).

\* All portions of the code written by Oxwall Foundation are Copyright (c) 2011. All Rights Reserved.

\* EXHIBIT B. Attribution Information

\* Attribution Copyright Notice: Copyright 2011 Oxwall Foundation. All rights reserved.

\* Attribution Phrase (not exceeding 10 words): Powered by Oxwall community software

\* Attribution URL: http://www.oxwall.org/

\* Graphic Image as provided in the Covered Code.

\* Display of Attribution Information is required in Larger Works which are defined in the CPAL as a work

\* which combines Covered Code or portions thereof with code not governed by the terms of the CPAL.

\*/

/\*\*

\* @author Nurlan Dzhumakaliev <nurlanj@live.com>

\* @package ow\_cron

\* @since 1.0

\*/

define('\_OW\_', true);

define('DS', DIRECTORY\_SEPARATOR);

define('OW\_DIR\_ROOT', substr(dirname(\_\_FILE\_\_), 0, - strlen('ow\_cron')));

define('OW\_CRON', true);

require\_once(OW\_DIR\_ROOT . 'ow\_includes' . DS . 'init.php');

OW::getPluginManager()->initPlugins();

$event = new OW\_Event(OW\_EventManager::ON\_PLUGINS\_INIT);

OW::getEventManager()->trigger($event);

//init cache manager

$beckend = OW::getEventManager()->call('base.cache\_backend\_init');

if ( $beckend !== null )

{

OW::getCacheManager()->setCacheBackend($beckend);

OW::getCacheManager()->setLifetime(3600);

OW::getDbo()->setUseCashe(true);

}

$plugins = BOL\_PluginService::getInstance()->findActivePlugins();

foreach ( $plugins as $plugin )

{

/\* @var $plugin BOL\_Plugin \*/

$pluginRootDir = OW::getPluginManager()->getPlugin($plugin->getKey())->getRootDir();

if ( file\_exists($pluginRootDir . DS . 'cron.php') )

{

include $pluginRootDir . DS . 'cron.php';

$className = strtoupper($plugin->getKey()) . '\_Cron';

$cron = new $className;

$runJobs = array();

$newRunJobDtos = array();

foreach ( BOL\_CronService::getInstance()->findJobList() as $runJob )

{

/\* @var $runJob BOL\_CronJob \*/

$runJobs[$runJob->methodName] = $runJob->runStamp;

}

$jobs = $cron->getJobList();

foreach ( $jobs as $job => $interval )

{

$methodName = $className . '::' . $job;

$runStamp = ( isset($runJobs[$methodName]) ) ? $runJobs[$methodName] : 0;

$currentStamp = time();

if ( ( $currentStamp - $runStamp ) > ( $interval \* 60 ) )

{

$runJobDto = new BOL\_CronJob();

$runJobDto->methodName = $methodName;

$runJobDto->runStamp = $currentStamp;

$newRunJobDtos[] = $runJobDto;

BOL\_CronService::getInstance()->batchSave($newRunJobDtos);

$newRunJobDtos = array();

$cron->$job();

}

}

}

}