```
public static function register($logName, $connectionString) {
  $urlData = parse_url($connectionString);

  if(! isset($urlData['scheme'])) {
    throw new Exception("Некорректная строка соединения $connectionString");
  }

  include_once('Logger/class.' . $urlData['scheme'] . 'LoggerBackend.php');

  $className = $urlData['scheme'] . 'LoggerBackend';
  if(!class_exists($className)) {
    throw new Exception('Для $urlData['scheme'] ' .
                        'нет базы данных');
  }

  $objBack = new $className($urlData);

  Logger::manageBackends($logName, $objBack);
}

public static function getInstance($name) {
  Logger::manageBackends($name);
}

private static function manageBackends($name, LoggerBackend $objBack = null) {
  static $backEnds;

  if(!isset($backEnds)) {
    $backEnds = array();
  }

  if(!isset($objBack)) {
    //
    if(isset($backEnds[$name])) {
      return $backEnds[$name];
    } else {
      throw new Exception("Заданная база данных $name" .
             "в журнале не зарегистрирована");
    }
  } else {
    $backEnds[$name] = $objBack;
  }
}
```

```
public static function levelToString($logLevel) {
  switch ($logLevel) {
    case LOGGER_DEBUG:
      return 'LOGGER_DEBUG';
      break;
    case LOGGER_INFO:
      return 'LOGGER_INFO';
      break;
    case LOGGER_NOTICE:
      return 'LOGGER_NOTICE';
      break;
    case LOGGER_WARNING:
      return 'LOGGER_WARNING';
      break;
    case LOGGER_ERROR:
      return 'LOGGER_ERROR';
      break;
    case LOGGER_CRITICAL:
      return 'LOGGER_CRITICAL';
```