|  |  |  |
| --- | --- | --- |
|  |  | УТВЕРЖДАЮ |
|  |  | Начальник тех.отдела |
|  |  | \_\_\_\_\_\_\_\_\_\_\_\_\_ Хххххх Х.Х. |
|  |  | “\_\_\_\_\_”\_\_\_\_\_\_\_\_\_\_\_\_200\_\_ |

**ФРЕЙМВОРК PHPUNIT**

**Руководство оператора**

**Лист утверждения**

**А.В.00001-01 34 01**

**А.В.00001-01 34 01-лу**

Инв. № подл.

Подпись и дата

Взам. инв. №

Инв. № дубл.

Подпись и дата

|  |  |  |
| --- | --- | --- |
|  |  | Руководитель разработки |
|  |  | Начальник ХХХХ |
|  |  | \_\_\_\_\_\_\_\_\_\_\_Хххххххх Х.Х. |
|  |  | “\_\_\_\_\_”\_\_\_\_\_\_\_\_\_\_\_\_200\_\_ |
|  |  |  |
|  |  | Ответственный исполнитель |
|  |  | Начальник гр. РиВ АСУТП ХХХХ |
|  |  | \_\_\_\_\_\_\_\_\_\_\_\_Ххххххх Х.Х. |
|  |  | “\_\_\_\_\_”\_\_\_\_\_\_\_\_\_\_\_\_200\_\_ |
|  |  |  |
|  |  | Исполнитель |
|  |  | Вед. инженер ХХХХ |
|  |  | \_\_\_\_\_\_\_\_\_\_\_\_\_Хххххх Х.Х. |
|  |  | “\_\_\_\_\_”\_\_\_\_\_\_\_\_\_\_\_\_200\_\_ |

**УТВЕРЖДЕНО**

А.В.00001-01 34 01-лу

**ФРЕЙМОВОРК PHPUNIT**

**Руководство оператора**

**Инв. № подл.**

**Подпись и дата**

**Взам. инв. №**

**Инв. № дубл.**

**Подпись и дата**

**А.В.00001-01 34 01**

# Аннотация

В данном программном документе приведено руководство оператора по применению и эксплуатации программы «PHPUnit», предназначенной для тестирования программ и создания удобной среды для программиста..

В данном программном документе, в разделе «Назначение программы» указаны сведения о назначении программы и информация, достаточная для понимания функций программы и ее эксплуатации.

В разделе «Условия выполнения программы» указаны условия, необходимые для выполнения программы (минимальный состав аппаратных и программных средств и т.п.).

В данном программном документе, в разделе «Выполнение программы» указана последовательность действий оператора, обеспечивающих загрузку, запуск, выполнение и завершение программы, приведено описание функций, формата и возможных вариантов команд, с помощью которых оператор осуществляет загрузку и управляет выполнением программы, а также ответы программы на эти команды.

В разделе «Сообщения оператору» приведены тексты сообщений, выдаваемых в ходе выполнения программы, описание их содержания и соответствующие действия оператора (действия оператора в случае сбоя, возможности повторного запуска программы и т.п.).

Оформление программного документа «Руководство оператора» произведено по требованиям ЕСПД (ГОСТ 19.101-77 [[1]](#footnote-1)1), ГОСТ 19.103-77 [[2]](#footnote-2)2), ГОСТ 19.104-78\* [[3]](#footnote-3)3), ГОСТ 19.105-78\* [[4]](#footnote-4)4), ГОСТ 19.106-78\* [[5]](#footnote-5)5), ГОСТ 19.505-79\* [[6]](#footnote-6)6), ГОСТ 19.604-78\* [[7]](#footnote-7)7)).

# Содержание

[Аннотация 2](#_Toc486237219)

[Содержание 3](#_Toc486237220)

[1. Назначение программы 4](#_Toc486237221)

[1.1. Функциональное назначение программы 4](#_Toc486237222)

[1.2. Эксплуатационное назначение программы 4](#_Toc486237223)

[1.3. Состав функций 5](#_Toc486237224)

[1.3.1. Функция тестирования операторов массива 5](#_Toc486237225)

[1.3.2. Функция использования вывода на экран для проверки операторов массива 5](#_Toc486237226)

[1.3.3. Функция тестирования операторов массива, сравнение ожидаемого результата и фактического значения 6](#_Toc486237227)

[1.3.4. Функция использования утверждения для тестирования оператора массива ………………………………………………………………………………....6](#_Toc486237228)

[1.3.5. Функция тестирования операций с массивами 6](#_Toc486237229)

[1.3.6. Функция использования аннотации @depends для описания зависимостей. 6](#_Toc486237230)

[1.3.7. Функция использования зависимостей между тестами 6](#_Toc486237231)

[1.3.8. Функция использования источника данных 6](#_Toc486237232)

[1.3.9. Функция использования аннотации @expectedException 6](#_Toc486237233)

[1.3.10. Функция применения @expectedException для ожидания ошибки PHP. 7](#_Toc486237234)

[1.3.11. Функция использование setup() для создания тестового окружения тестирования стека 7](#_Toc486237235)

[2. Условия выполнения программы 7](#_Toc486237236)

[2.1. Минимальный состав аппаратных средств 7](#_Toc486237237)

[2.2. Минимальный состав программных средств 8](#_Toc486237238)

[2.3. Требования к персоналу (пользователю) 8](#_Toc486237239)

[3. Выполнение программы 8](#_Toc486237240)

[3.1. Загрузка и запуск программы 8](#_Toc486237241)

[3.2. Выполнение программы 8](#_Toc486237242)

[3.3. Завершение работы программы 9](#_Toc486237243)

[4. Сообщения оператору 9](#_Toc486237244)

[4.1. Сообщение о отсутствии файла PHPUnit 9](#_Toc486237245)

[Лист регистрации изменений 10](#_Toc486237246)

# Назначение программы

## **Функциональное назначение программы**

Фрейм­ворк – это про­грамм­ное обес­пе­че­ние, облег­ча­ю­щее раз­ра­ботку и объ­еди­не­ние раз­ных ком­по­нен­тов боль­шого про­грамм­ного про­ек­та. Эта платформа под­хо­дит для созда­ния сай­тов, биз­нес-при­ло­же­ний и веб-сер­ви­сов.

Фрейм­ворк отли­ча­ется от биб­лио­теки тем, что биб­лио­тека может быть исполь­зо­вана в про­грамм­ном про­дукте про­сто как набор под­си­стем близ­кой функ­ци­о­наль­но­сти, не влияя на архи­тек­туру основ­ного про­грамм­ного про­дукта и не накла­ды­вая на неё ника­ких огра­ни­че­ний. Фрейм­ворк дик­тует пра­вила постро­е­ния архи­тек­туры при­ло­же­ния, зада­вая на началь­ном этапе раз­ра­ботки пове­де­ние по умол­ча­нию, фор­ми­руя кар­кас, кото­рый нужно будет рас­ши­рять и изме­нять согласно ука­зан­ным тре­бо­ва­ни­ям. Фрейм­ворк может вклю­чать вспо­мо­га­тель­ные про­грам­мы, биб­лио­теки кода, язык сце­на­риев и дру­гое ПО, облег­ча­ю­щее раз­ра­ботку и объ­еди­не­ние раз­ных ком­по­нен­тов боль­шого про­грамм­ного про­ек­та.

Одним из глав­ных пре­иму­ществ при исполь­зо­ва­нии фрейм­вор­ков явля­ется то, что веб-при­ло­же­ния часто исполь­зуют стан­дар­ти­зи­ро­ван­ную струк­туру орга­ни­за­ции ком­по­нен­тов. Сооз­да­ние струк­туры при раз­ра­ботке на фрейм­вор­ках очень упро­ща­ет­ся. По сути, фрейм­ворк — это мно­же­ство кон­крет­ных и абстракт­ных клас­сов, а также опре­де­ле­ний спо­со­бов их вза­и­мо­от­но­ше­ния. Кон­крет­ные классы обычно реа­ли­зуют вза­им­ные отно­ше­ния между клас­са­ми, а абстракт­ные классы пред­став­ляют собой точки рас­ши­ре­ния, в кото­рых кар­касы могут быть исполь­зо­ваны или адап­ти­ро­ва­ны. Для обес­пе­че­ния рас­ши­ре­ния воз­мож­но­стей обычно исполь­зуются тех­ники объ­ектно-ори­ен­ти­ро­ван­ного про­грам­ми­ро­ва­ния (напри­мер, части при­ло­же­ния могут насле­до­ваться от базо­вых клас­сов фрейм­ворка). Программа «PHPUnit» работает под управлением ОС Windows 7 или версии выше.

## **Эксплуатационное назначение программы**

Тестирования кода и создания удобной среды для программиста.

## **Состав функций**

1)Тестирование операторов массива.

2)Использование вывода на экран для проверки операторов массива.

3)Тестирование операторов массива, сравнение ожидаемого результата и фактического значения.

4)Использование функции утверждения для тестирования операторов массива.

5)Тестирование операций с массивами.

6)Использование аннотации @depends для описания зависимостей.

7)Использование зависимостей между тестами.

8)Использование источника данных.

9)Использование аннотации @expectedException.

10)Применение @expectedException для ожидания ошибки PHP.

11)Использование метода setup() для создания тестового окружения тестирования стека.

### Функция тестирования операторов массива

На одном из этапов тестирования надо проверить работу метода count(). Мы ожидаем, что для только что созданного массив, метод count() вернет 0. После добавления элемента в массив count() должна вернуть 1.

### Функция использования вывода на экран для проверки операторов массива

Самый простой способ проверить, получили ли мы то, что хотели — вывести результат работы метода count() на экран. Если мы сначала получили 0 а потом 1, то array и count() работают корректно.

### Функция тестирования операторов массива, сравнение ожидаемого результата и фактического значения

 Добавим в тест сравнение ожидаемого результата и фактического значения, выведем ok если ожидаемый и фактический результаты совпали.  
Если вывод будет not ok, значит где-то произошла ошибка.

### Функция использования утверждения для тестирования оператора массива

Тест генерирует сообщение только в случае ошибки.

### Функция тестирования операций с массивами

Наименования тестирующих методов образуются путем добавления приставки test к наименованиям тестируемых методов.

### Функция использования аннотации @depends для описания зависимостей.

Первый тест, метод testEmpty(), создает пустой массив и задает утверждение, что массив пустой. После этого тест возвращает окружение (fixture) в качестве результата.   
Второй тест, testPush(), зависит от testEmpty() и получает результат работы testEmpty() в качестве аргумента. И наконец, метод testPop() зависит от testPush().

### Функция использования зависимостей между тестами

Улучшение обнаружения дефектов достигается за счет использования зависимостей между тестами. У теста может быть несколько аннотаций @depends.  
PHPUnit не изменяет последовательность, в которой запускаются тесты.

### Функция использования источника данных

Метод источника данных должен быть public и должен возвращать массив массивов или объект, поддерживающий интерфейс Iterator, который на каждой итерации возвращает массив.  
Для каждого массива, который является частью коллекции, будет вызван тестовый метод. В качестве аргумента методу будет передан массив значений.

### Функция использования аннотации @expectedException

Первый тест, testEmpty(), создает пустой массив и задает утверждение, что массив пустой. После этого тест возвращает окружение (fixture) в качестве результата.   
Второй тест, testPush(), зависит от testEmpty() и получает результат работы testEmpty() в качестве аргумента. И наконец, testPop() зависит от testPush().

### Функция применения @expectedException для ожидания ошибки PHP.

По умолчанию, ошибки, предупреждения и уведомления PHP, которые появляются во время тестирования, PHPUnit преобразует в исключения. Используя эту особенность, Вы можете, настроить механизм ожидания появления подобного исключения в тесте

### Функция использование setup() для создания тестового окружения тестирования стека

Шаблонные методы setUp() и tearDown() вызываются по одному разу для каждого тестового метода (и для нового экземпляра) тестового класса. Шаблонные методы setUpBeforeClass() и tearDownAfterClass() вызываются прежде, чем будет выполнен первый метод тестового класса и после того как последний метод будет завершен.

# Условия выполнения программы

## **Минимальный состав аппаратных средств**

Минимальный состав используемых технических средств:

* 32-разрядный (x86) или 64-разрядный (x64) процессор\* с тактовой частотой 1 ГГц или выше.
* 1 ГБ (для 32-разрядного процессора) или 2 ГБ (для 64-разрядного процессора) ОЗУ.
* 16 ГБ (для 32-разрядной системы) или 20 ГБ (для 64-разрядной системы) свободного места на жестком диске.

## **Минимальный состав программных средств**

Системные программные средства, используемые программой PHPUnit, должны быть представлены локализованной версией операционной системы Windows 7 или выше.

Также для функционирования программы PHPUnit на ПК необходимо предустановленное программное обеспечение стороннего разработчика, программа NetBeans IDE 8.2.

## **Требования к персоналу (пользователю)**

Пользователь должен владеть на уверенном уровне языком программирования PHP.

# Выполнение программы

## **Загрузка и запуск программы**

* Запустить NetBeans IDE 8.2 и создать проект PHP.
* Подключить PHPUnit введя в начала программы, код подключения фреймворк “require\_once 'PHPUnit/Framework.php';<br>”.

## **Выполнение программы**

Написать тест для проверки работы программы PHPUnit в php проекте NetBeans IDE 8.2 и запустить проверку.

## **Завершение работы программы**

Закрыть NetBeans IDE 8.2 при необходимости сохранив проект.

# Сообщения оператору

## **Сообщение о отсутствии файла PHPUnit**

Консоль «NetBeans IDE 8.2» выдаст сообщение об ошибке при отсутствии phar архива PHPUnit.phar.

ПРИЧИНА. Указанный в сообщении phar архив (PHPUnit.phar) не найден в проекте, или данный каталог не существует.

ДЕЙСТВИЯ ПРОГРАММЫ. Программа NetBeans IDE 8.2 не производит подключение фреймворка PHPUnit, проверки кода не происходит.

ДЕЙСТВИЯ ПОЛЬЗОВАТЕЛЯ. Проверить по указанному пути наличие фреймворка PHPUnit и в случае необходимости сообщить об ошибке системному администратору.

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Лист регистрации изменений | | | | | | | | | |
| Номера листов (страниц) | | | | | Всего  листов  (страниц)  в докум | №  документа | Входящий  № сопрово  дительного  документа  и дата | Подп. | Дата |
| Изм | изменен  ных | заме  ненных | новых | анулиро  ванных |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |

1. 1) ГОСТ 19.101-77 ЕСПД. Виды программ и программных документов [↑](#footnote-ref-1)
2. 2) ГОСТ 19.103-77 ЕСПД. Обозначение программ и программных документов [↑](#footnote-ref-2)
3. 3) ГОСТ 19.104-78\* ЕСПД. Основные надписи [↑](#footnote-ref-3)
4. 4) ГОСТ 19.105-78\* ЕСПД. Общие требования к программным документам [↑](#footnote-ref-4)
5. 5) ГОСТ 19.106-78\* ЕСПД. Общие требования к программным документам, выполненным печатным способом [↑](#footnote-ref-5)
6. 6) ГОСТ 19.505-79\* ЕСПД. Руководство оператора. Требования к содержанию и оформлению [↑](#footnote-ref-6)
7. 7) ГОСТ 19.604-78\* ЕСПД. Правила внесения изменений в программные документы, выполненные печатным способом [↑](#footnote-ref-7)