Slide 1:

Dobry den, dnes chcel by Vam povedat o kniznice PHPUnit.

Slide 2:

Nasledujuca agenda

- Preco pouzivame unit-tesovanie

- Co ze PHPUnit, ako ho nainstalovat’

- Pozrieme jednoduchy priklad, pripady uspesneho resp. neuspesneho testu

- Pozname co su Asserty a Výnimky

- Ako mozem testovat vystup funkcie

- Pozreme na viac-menej komplikavany priklad

- Urobime nejake vysledky

Slide 3:

Zacneme, unit-testovanije je urcene pre podielu vasho programu na vela malych kusov, a testovanije kazdej oddelnej casti.

To je dobre pre zastrešeniu Vasho kodu.

Ale pisanie testos zobera priblizne 20-30% od celkoveho casu vyvoju. Aj preto vacsina ludi vobec vynecháju tentu cast vyvoja.

Slide 4:

Ale pozrieme sa na tento priklad - zrejme odpoved bude 8. Ale sme nepozname toto dokedy nespustime skript.

V podstate, kedy by sme pouzivali kniznicu BCMath to mali by sme inu odpoved.

(ukazat priklad praci bcmath, zdroj

http://vmxpotapov.fei.stuba.sk/testing/bcmath-example/bcmath-example.php)

Slide 5:

Plusy testovania su:

problemy kodu sa nachadzaju skor.

testovanie dobre pre velkych projektov

Minusy:

Samozrejme zobera viac casu

Ale prepisom testov chyby nezmiznú

Slide 6:

Tu su 2 kniznicy pre testovania - PHPUnit a SimpleTest. Ale vsetcy pouzivaju PHPUnit

Slide 7:

Instalovanie mozete pozriet na tomto slide. Pojdem dalej

Slide 8:

Jednoduchy priklad - pocitanije mocniny cisla.

(ukazat priklad <https://github.com/braveS/phpUnitTesting/tree/master/class1/MyClass.php> a <https://github.com/braveS/phpUnitTesting/tree/master/class1/MyClassTest.php>)

Slide 9:

Spusteme test prikazom … vidime odpoved - bodka - znaci ze test bol spusteny uspesne; cas, ak dlho pracoval test; kolko pamati zobral; OK - znaci uspesny test, v zatvorkach mame pocet testovacych metodov, a kolvo krat boli spusteny metody.

Slide 10:

Niejaka informacia.

Meno testovacej triedy sa sklada s slova Test a mena testujucej triedy.

Testujuca trieda dedi PHPUnit\_Framework\_TestCase

Kazdy test je publik-metodom, meno tiez sa sklada zo slova test a mena metodu.

V kazdej metode pouzivame assert-metod pre vyhľadania, cize je vysledok testu rovnaky vysledku ktory my chceme.

Slide 11:

Data providery,

Pouzivame, kedy chceme odoslat metodu subor zoznamov vchodnych udayov.

Pouzivame s pomocou @dataProvider zapisu

(ukazat neuspesny priklad <https://github.com/braveS/phpUnitTesting/blob/master/class1/MyClassTestFail.php>)

Slide 12:

Asserty pouzivame pre kazdej testovacej situacije. Napriklad sme pouzili assertEquals() kedy chceli porovnat vysledok funkcije.

Tu su viac assertov pre roznych pripadov.

Slide 13:

Ano, Chuck Norris pouziva len jeden assert)

Slide 14:

Dobry kod sa pouziva Výnimky. Ano my mozme ih testovat.

(ukazat priklad <https://github.com/braveS/phpUnitTesting/tree/master/class2/MyClass2.php> a <https://github.com/braveS/phpUnitTesting/tree/master/class2/MyClass2Test.php>)

Slide 15:

Testujeme output funkcie

(ukazat priklad <https://github.com/braveS/phpUnitTesting/blob/master/class3/MyClass3Test.php>)

Slide 16:

Dobre, mozeme pozriet testovanie prikladu algoritmu, ktory sme vyvoali na prvom testu.

ukazat zdroj <https://github.com/braveS/phpUnitTesting/tree/master/class4/MyClass4.php> a <https://github.com/braveS/phpUnitTesting/tree/master/class4/MyClass4Test.php>)

Slide 17:

PHPUnit moze testovat databazu, este mozeme pouzivat moc-objekty, pisat analitiku do kodu, a viac ineho..

Slide 18:

Namesto vysledku chcel b ypovedat, ze pouzivanie unit-testovania neni striebnej gulej, chyby budu, ale ich pocet bude uz menej… Lepsie pisat testy.

Slide 19:

Toto su zdroje

Slide 20:

Dobre, vtedy dakujem za pozor!