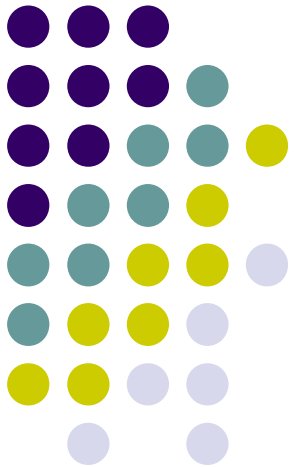
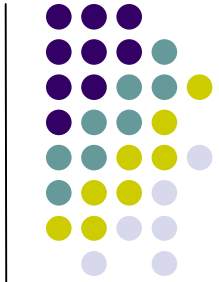
PHP

Autorid:
Aleksandr Vaskin
Aleksandr Bogdanov



Keelest



- Skriptikeel – skript teeb oma tööd pärast seda, kui toimus mingi sündmus*
- Orienteeritud programmeerija eesmärkide saavutamiseks (mugavus on tähtsam kui vastavus standartidele)
- Serveripoolne keel
- Platvormist sõltumatu*
- Saab kasutada nii HTMLi sees (*HTML embedded*), kui ka eraldiseisvana skriptina

PHP ajalugu I

- Aastal 1994 taani programmeerija

  Rasmus Lerdorf kirjutas skriptide kogumi

  Perl keeles selleks, et pidada arvet tema online resumee külastajatest, mis töödeldasid HTML dokumentide šabloone. Skriptide kogumi nimeks sai: ***P**ersonal **H**ome**p**age Tools*
- Aastal 1997 ilmus teine interpretaatori versioon kirjutatud C keeles – PHP/FI 2.0. Mida kasutasid liigikaudu 1% kõikidest internet domeenidest maailmas.

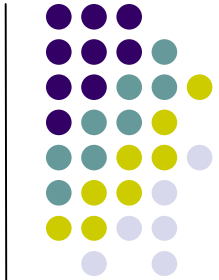
PHP ajalugu II

- 1997 – Andi Gutmans ja Zeev Suraski, kaks arendajat israili tehnoloogia instituudist (Technion) kirjutasid koodi ümber nullist, kuna pidasid PHP/FI 2.0 kõlbamtuks elektroonilise kommertsi rakenduste arendamiseks.

  Koostöö tegemiseks PHP 3.0 kallal Andi, Rasmus ja Zeev otsustasid ühilduda ja kuulutada PHP 3.0 ametlikuks PHP/FI järglaseks.

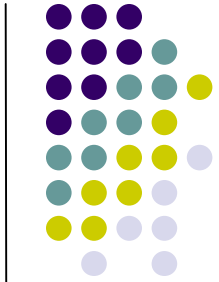
  PHP/FI areng oli selleks ajaks praktiliselt peatatud.

PHP ajalugu III



- Juuni 1998 – PHP 3.0 oli ametlikult väljalastud. Aasta lõpuks PHP 3.0 oli installeeritud ~10% veebiserveritel. PHP 3.0 võimaldas tuuma laiendamist, uus võimsam ja täielikum süntaks OOP toega. PHP 3.0 sai ka uue nime – ”PHP: Hypertext Preprocessor”.
- 1998 talveks hakkasid Andi Gutmans ja Zeev Suraski PHP tuuma ümbertöötamist.

PHP ajalugu IV



- Mai 2000 - PHP 4.0

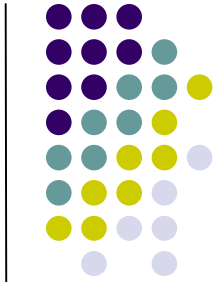
  Uus tuum **Ze**[ev][A]**nd**[i] **Engine**, mille eesmärkideks olid:

  1. keeruliste rakenduste jõudsuse tõstmine
  2. PHP baasise koodi modulaarsuse parandamine

  Lisaks:

  1. sessioonide toetust
  2. sisendi puhverdamine
  3. paar uut keele konstruktsiooni
  4. turvalisemad meetodid kasutaja sisendi töötlemiseks.

PHP ajalugu V

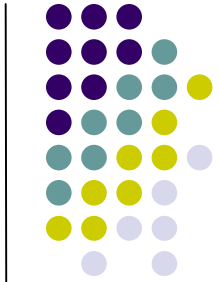


- 13 juuli 2004 – PHP 5.0

  Täielikult ümbertöödetud OOP funktsioonid (lisati destruktor, objektide kloonimine, abstraktsed klassid, privaat ja kaitstud meetodid ja muutujad) uuendused olid tehtud kõrge ühitatavuse säilitamise arvestamisega.
- PHP 6 - tuumast planeeritakse eemaldada:
  1. POSIX regulaaravaldised(mallid)
  2. “pikad” superglobaalsed massiivid
  3. Magic Quotes

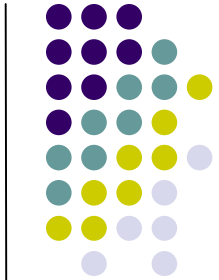
  Planeeritakse teha parem unicode tugi

PHP süntaks



- Kõik PHP kood kirjutatakse **<?php** ja **?>** tag'ide vahele.
- Iga koodi rida tuleb lõppetada "**;**" märgiga.
- Teksti väljastamiseks on kaks põhiavaldust:
  1. **echo**
  2. **print**
- Muutujad
  - algavad **$** märgiga, nt.: `$muutuja1`
  - muutuja nimi peab algama kirjatähega või alamkriipsuga;
  - muutuja võib koosneda ainult kirjatähtedest, numbritest ja alamkriipsutest;
  - Tõusutundlikkus: `$muutuja`≠`$Muutuja`≠`$MuuTujA`

PHP süntaks II



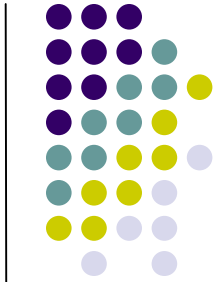
- Kommentaarid:

```
1. //üherealise kommentaari algus
2. /*
     mitmerealise kommentaari
     blokk
   */
3. #saab kah “#” kasutada, aga on vananenud
```

“Hello World” näide:

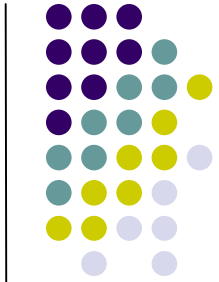
```
<?php
//Hello World näide
echo “Hello World”;
?>
```

PHP süntaks III - operaatorid



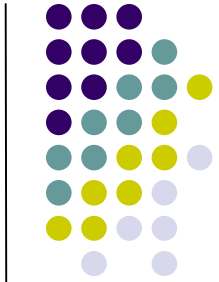
Operaator	Kirjeldus	Näide	Resultaat
+	Liitmine	x=2 x+2	4
-	Lahutamine	x=2 5-x	3
*	Korrutamine	x=4 x*5	20
/	Jagamine	15/5	3
%	Mooduli leidmine (jagamise jääk)	5%2	1
++	Suurendamine 1 võrra	x=5 x++	6
--	Kahanemine 1 võrra	x=5 x--	4

PHP süntaks IV – omistamise operaatorid



Operaator	Näide	On sama nagu
=	x=y	x=y
+=	x+=y	x=x+y
-=	x-=y	x=x-y
*=	x*=y	x=x*y
/=	x/=y	x=x/y
%=	x%=y	x=x%y

PHP süntaks V – võrdlemise - ja loogilised operaatorid



- == võrdub
- != ei võrdu
- > on suurem
- < on väiksem
- >= on suurem või võrdne
- <= on väiksem või võrdne
- === range võrdlus (tüübi kontrolliga)

- && ja
- || või
- ! eitus
- Lisaks veel bitikaupa tehete (*bitwise*) operaatorid

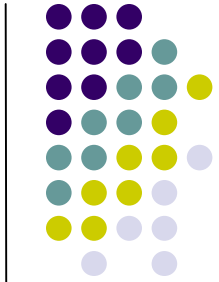
PHP süntaks VI – tingimusavaldised

- if (*tingimus*){
  kood mida täidetakse siis kui tingimus on tõene;
  ...;
  }

else{
  kood mida täidetakse siis kui tingimus on väär;
  ...;
  }

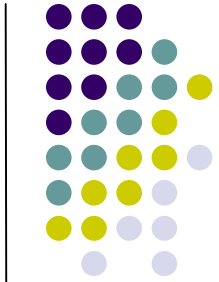
- *ANSI/ISO C stiilis: (avaldis) ? tõene : väär;*

PHP süntaks VII – tingimusavaldised



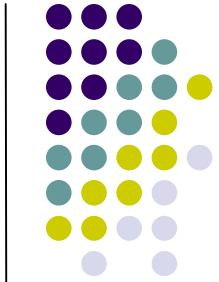
- *switch (avaldis){*
  case etikett1:
  kood mida täidetakse siis kui avaldis = etikett1;
  break;
  case etikett2:
  kood mida täidetakse siis kui avaldis = etikett2;
  break;
  default:
  kood mida täidetakse siis kui avaldis erineb nii etikett1´st kui ka etikett2´st;
  }

PHP süntaks VIII – avaldised



- Lisaks on sellised avaldised nagu:
  1. for
  2. foreach
  3. while
  4. do-while
  5. require()/require_once()
  6. include()/include_once()
  7. ...

PHP süntaks IX – funktsioonid

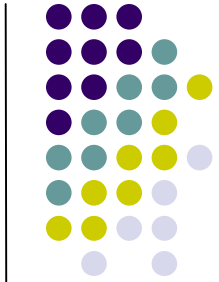


- Kasutaja defineeritud
- Sisemised (sisseehitatud) funktsioonid

Kasutaja funktsioonide näide:

```
<?php
    function f($arg_1, $arg_2, /* ..., */
$arg_n)
{
    echo “Funktsiooni näidis.\n";
    return $retval;
}
?>
```

PHP süntaks X – *funktsioonid*

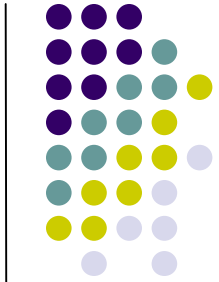


- Rekursiooni näide:

```
<?php
function factorial( $n ){
  if ($n == 0 || $n== 1){
     return 1;
  }
  else{
     return ( $n * factorial ($n-1));
  }
}
?>
```

- Sisemised funktsioonid:
  - phpinfo();
  - print_r();
  - …

PHP süntaks XI – muutujad



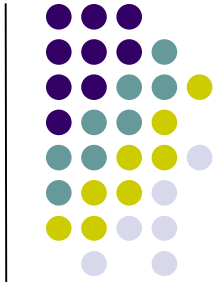
- Muutujate tüübikontroll ei ole range, teisendamine on automaatene.
- Konstandid:

```
define(“PI”, 3.14159);
```

- globaalsed vs. lokaalsed muutujad
- PHP välised muutujad nt. vormilt:

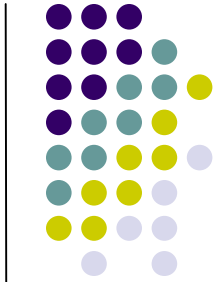
```
<form action="foo.php" method="post">
Name: <input type="text" name="username"/><br/>
Email: <input type="text" name="email"/><br/>
<input type="submit" name="submit"
value="Submit!"/>
</form>
```

PHP süntaks XII – muutujad



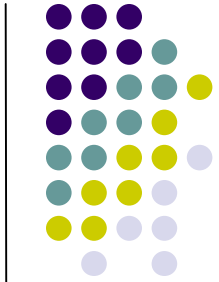
```php
<?php
  // alates PHP 4.1.0
  echo $_POST['username'];
  echo $_REQUEST['username'];
  echo $HTTP_POST_VARS['username'];
?>
```

PHP süntaks XIII – muutujad



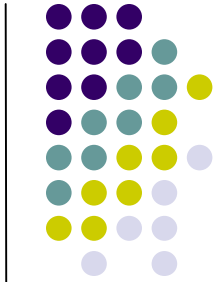
- Massiivid:
```
$Massiiv=('RED' => '#FF0000', 2006,
$Massiiv_2("See","on","kah","massiiv"));
```
  - Massiivid on heterogeensed – üks massiiv saab sisaldada mitmeid tüüpe objekte.
  - Massiivid võivad olla nii assotiatiivsed kui ka korrastatud.
- Sõned e. Strings:
  - Tekst "" või '' märkide vahel.
  - Muutujad lubatud "" märkide vahel.
  - Sõnede liitmine "." abil
```
$String = "Mingi tekst";
$String .= " käib siia";
echo $String;
```
    Tulemus:
```
Mingi tekst käib siia
```

PHP süntaks XIV – muutujad



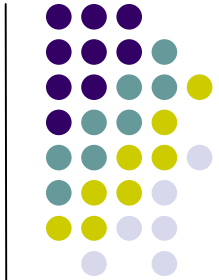
- Booli tüüp
  Kaks võimaliku väärtust – true (tõene) või false (väär). Saab kasutada ka C keele süntaksi – nullist erinev väärtus == true (samuti ka mittetühi sõne), 0 == false (samuti tühi sõne).
- Täisarvud e. Integer:
  Täisarvu tüüp hoiab numbreid platvormi-sõltuvas piirkonnas, tüüpiliselt 32-bitti märkidega täisarv. Täisarvu muutujad saab kasutada nii heksa, oktaal kui ka kümnend notatsioonis.

PHP süntaks XV



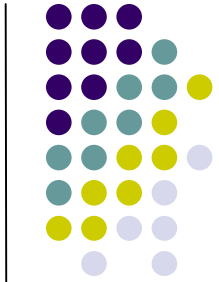
- Ujukoma arvud:
  - tavalisele notatsioon: $ujukoma_arv = 1.234;
  - teadusliku notatsioon: $ujukoma_arv = 1.2E3
- Null – väärtuseta muutuja.

PHP süntaks XVI – erandid



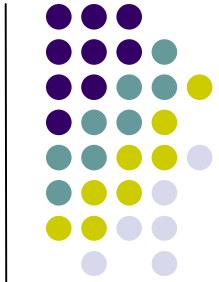
- Alates 5. versioonist
- Erandite mudel on sarnane teiste programmeerimiskeeltega (nt. Java). Erandit saab visata (*throw*), ja püüda (*catch*) PHP koodi sees. Kood võib olla ümbritsetud *try* blokkiga, potentsiaalse viga avastamise kergendamiseks. Igal *try* avaldisel peab olema vähemalt üks vastav *catch* blokk. Mitut *catch* blokki saab kasutada erinevate erandite püüdmiseks. Tavaline täitmine (kui erandeid ei viska *try* blokki sees, või siis kui *catch* vastav visatud erandi klassile ei leidu) täidetakse edasi pärast viimast *catch* blokk´i defineeritud sarjas. Erandeid võib visata (või korduvalt visata) *catch* blokki sees.

PHP süntaks XVII – erandid



- Kui erandit püütakse, koodi mis järgneb avaldisele enam ei täideta, ja PHP üritab otsida esimest vastava *catch* blokki. Kui erand pole püütud, siis väljastatakse PHP Fatal Error koos "*Uncaught Exception* ..." sõnumiga.

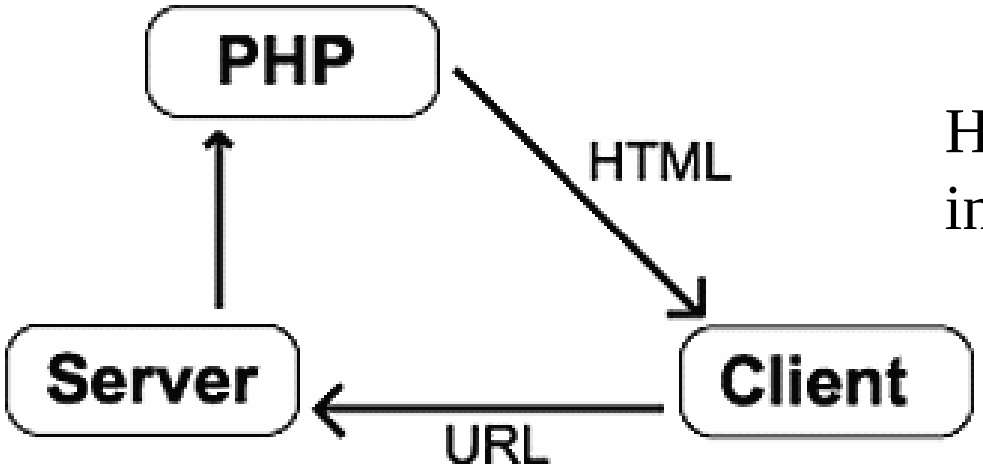
PHP ja OOP



- PHP 3,4 – OOP on võimalik, kuid ei vasta täielikult paradigmale.
- PHP 5 – nüüd on olemas muutujate nähtavuse piirajad (public, protected, private), konstruktorid, destruktorid, liidesed.

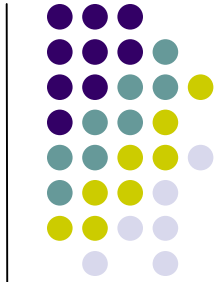
Kuidas PHP töötab

Tavalise HTTP päringu töötlus



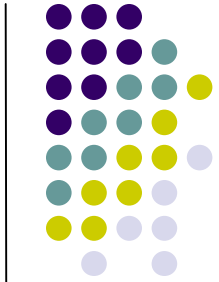
HTTP päringu töötlus koos PHP interpretaatoriga

PHP plussid

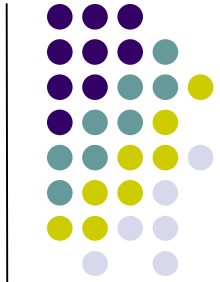


- Lihtne õppida
- Keel pidevalt areneb*
- Palju näiteid ja valmiskoodi
- Palju laiendusi nt.:
  - erinevate andmebaaside tugi
  - DOM
  - piltide töötlus
  - Krüpteerimine
  - andmete pakkimine
  - XML
  - mailide saatmine

PHP miinused



- Ei nõuta muutujate deklareerimist enne muutujate väärtustamist.
- Muutujad pole piiratud oma tüübiga – muutujat $i saab väärtustada täisarvuga, siis väärtustada sõnega ning pärast veel väärtustada andmete massiiviga
- Funktsioonide/meetodite ülelaadimine pole lubatud
- Tüübi kontroll pole range! (tuleb kasutada ===)



Täname!