* **Versões do PHP**
  + **1995**
    - **PHP 1.0**
      * Personal Home Page
      * Funcionalidades simples para um site básico.
      * Não tinha características de linguagem em si.
  + **1997**
    - **PHP 2.0**
      * Oficialmente nomeado PHP/FI 2.0.
      * Linguagem standalone.
      * Recursos limitados.
  + **1998**
    - **PHP 3.0**
      * Primeira versão colaborativa.
      * Características mais relacionadas a uma linguagem.
      * Extension API.
  + **2000**
    - **PHP 4.0**
      * Zend Engine 1.0.
      * Melhoria de performance.
      * Aumento da modularização.
      * Uso de super globais $\_GET, $\_POST, $\_SESSION, etc.
  + **2004**
    - **PHP 5.0**
      * Zend Engine 2.0.
      * Primeira versão com Orientação a Objetos.
      * PHP Data Objects (PDO).
      * Operador de exponenciação.
      * Suporte a JSON.
      * Namespaces, closures, garbage collection, exception handling
      * Novo depurador (phpdbg)
  + **2005**
    - **PHP 6.0**
      * Aumento da segurança e performance.
      * Suporte nativo a Unicode.
      * Até existiram algumas versões.
      * Nunca recebeu uma release oficial.
  + **2015**
    - **PHP 7.0**
      * Zend Engine 3.0.
      * Performance até 9 vezes maior.
      * Escape de códigos Unicode
      * Operador null coalescing ??
      * Declaração escalar.
      * Operador spaceship <=>
      * Classes anônimas.
      * Retorno de void.
      * Libsodium
      * Foreign function interface
  + **2020**
    - **PHP 8.0**
      * Zend Engine 4.0.
      * Just-In-Time compilation.
      * Locale-Independent number convertions.
      * Named arguments.
      * Expressão match.
      * Operador nullsafe.
      * Enumerations.
      * Readonly properties.
      * Fibers.
* Regras para nomes identificadores:
  + Variáveis sempre começam com o símbolo $
  + O segundo pode ser letra ou o símbolo \_
  + Aceita caracteres [a-z], [A-Z], [0-9] e [\_]
  + Aceita caracteres da tabela ASCII a partir de 128
  + Aceita caracteres acentuados como à, õ, ç
  + A linguagem é case sensitive em relação aos nomes
  + Nomes especiais como $this não podem ser usados
* Recomendações para dar nomes:
  + Tente dar nomes claros e de fácil identificação
  + Evite nomes muito curtos ou muito longos
  + Defina um padrão e siga em todo o projeto
  + Para variáveis, dê preferência a letras minúsculas
  + Para constantes, dê preferência a letras maiúsculas
  + Use camelCase para métodos e atributos
  + Use SNAKE\_CASE para nomear constantes