**Radar Criminal**

* **Pacote: controller**
  + Classe: CategoriaController.php
    - Função: \_listarTodas
      * Variável: arrayCategoria = recebe lista de todas as categorias
    - Função: \_consultarPorId
      * Variável: id = número de identificador
      * Variável: categoria = recebe a categoria da consulta pelo id
  + Classe: CrimeController.php
    - Função: \_consultarPorId
      * Variável: id = número de identificador do crime
    - Função: \_consultarIdNatureza
      * Variável: natureza = recebe a natureza do crime consultada pelo id
    - Função: \_inserirCrime
      * Variável: crime = recebe crime informado pelo usuário
    - Função: \_somaDeCrimePorNatureza
      * Variável: natureza = recebe natureza do crime
    - Função: \_somaDeCrimePorNaturezaEmAno
      * Variável: natureza =
      * Variável: ano =
    - Função: \_somaDeCrimePorano
      * Variável: ano =
    - Função: \_somaCrimeTodosAnos
      * Variável: ano =
      * Variável: somaTodasAnos =
  + Classe: NaturezaController.php
  + Classe: RegiaoAdministrativaController.php
    - * Variável: raDAO = variável usada para instanciar um objeto da classe RegiaoAdmnistrativaDAO
    - Função: \_\_construct()
      * Variável: raDAO
    - Função: \_listarTodas()
      * Variável: raDAO
      * Variável: arrayRA = vetor que contém todas as regiões administrativas
  + Classe: TempoController.php
    - * Variável: tempoDAO
    - Função: \_\_construct
    - Função: \_listarTodos
      * Variável: tempoDAO
    - Função: \_listarTodasEmOrdem
      * Variável: tempoDAO
    - Função: \_\_constructTeste
      * Variável: tempoDAO
    - Função: \_consultarPorId
      * Variável: tempoDAO
      * Variável: id
    - Função: \_consultarPorIntervalo
      * Variável: tempoDAO
      * Variável: intervalo
    - Função: \_inserirTempo
      * Variável: tempoDAO
      * Variável: tempo
    - Função: \_inserirTempoArrayParse
      * Variável: arrayTempo
      * Variável: i
      * Variável: dadosTempo
      * Variável: tempoDAO
    - Função: \_inserirTempoArrayParseQuadrimestral
      * Variável: arrayTempo
      * Variável: i
      * Variável: arrayAno
      * Variável: ano
      * Variável: dadosTempo
      * Variável: j
      * Variável: arrayDadosTempo
    - Função: \_retornarDadosFormatados
      * Variável: dadosTempo
      * Variável: arrayDadosTempo
      * Variável: i
      * Variável: dados
* **Pacote: model**
  + Classe: Categoria.php
    - * Variável: idCategoria = tem o id da categoria de crimes
      * Variável: nomeCategoria = nome da categoria de crime
    - Função: \_\_setIdCategoria($idCategoria)
      * Variável: idCategoria
    - Função: \_\_getIdCategoria
      * Variável: idCategoria
    - Função: \_\_setNomeCategoria($nomeCategoria)
      * Variável: nomeCategoria
    - Função: \_\_getNomeCategoria()
      * Variável: nomeCategoria
    - Função: \_\_constructOverload ($idCategoria, $nomeCategoria)
      * Variável: idCategoria
      * Variável: nomeCategoria
    - Função: \_\_construct()
  + Classe: Crime.php
  + Classe: Natureza.php
  + Classe: RegiaoAdministrativa.php
* Variável: idRegiaoAdministrativa: é o identificador da região administrative. Deve ser um valor numérico.
* Variável: nomeRegiao: é o nome da região administrativa. Deve ser uma string.
* Função: \_\_construct()
* Função: \_\_constructOverLoad($idRA,$nomeRegiao)
* Variável: idRA = tem o valor que será passado à variável idRegiaoAdministrativa.
* Variável: nomeRegiao = contém a string que será passada para a variável nomeRegiao.
* Função: \_\_setIdRegiaoAdministrativa($idRegiaoAdministrativa)
  + - * Variável: idRegiaoAdministrativa
* Função: \_\_getIdRegiaoAdministrativa()
* Função: \_\_setNomeRegiao($nomeRegiao)
  + - * Variável: nomeRegiao
* Função: \_\_getNomeRegiao()
  + Classe: Tempo.php
    - * Variável: idTempo = identificador de tempo
      * Variável: intervalo = recebe o valor do intervalo
      * Variável: mes = indica o mes
    - Função: \_\_setIdTempo
      * Variável: idTempo
    - Função: \_\_getIdTempo
      * Variável: idTempo
    - Função: \_\_setIntervalo
      * Variável: intervalo
    - Função: \_\_getIntervalo
      * Variável: intervalo
    - Função: \_\_setMes
      * Variável: mes
    - Função: \_\_getMes
      * Variável: mes
    - Função: \_\_construct
    - Função: \_\_constructuOverload
      * Variável: idTempo
      * Variável: intervalo
      * Variável: mes
* **Pacote: persistence**
  + Classe: CategoriaDAO.php
    - * Variável: conexao = variável usada pra instanciar um objeto da classe Conexao pra estabelecer conexão com o banco de dados.
    - Função: \_\_construct()
      * Variável: conexao
    - Função: \_\_constructTeste()
      * Variável: conexao
    - Função: \_\_constructTeste()
      * Variável: conexao
    - Função: listarTodas()
      * Variável: sql = usada pra acessar a tabela regiao\_administrativa;
      * Variável: resultado = recebe o resultado, se foi bem sucedida, da seleção da tabela regiao\_administrativa;
      * Variável: dadosRA = instancia um objeto da classe RegiaoAdministrativa para ter acesso aos dados (nome e id) de uma região administrativa específica;
      * Variável: retornaRAs[] = vetor que contém a listagem de todas as regiões administrativas.
      * Variável: dadosRA = instancia um objeto da classe RegiaoAdministrativa para ter acesso aos dados (nome e id) de uma região administrativa específica.
      * Variável: retornaRAs[] = vetor que contém a listagem de todas as regiões administrativas
    - Função: listarTodasAlfabeticamente()
      * Variável: sql = usada pra acessar a tabela regiao\_administrativa ordenada alfabeticamente
      * Variável: resultado
      * Variável: dadosRA
      * Variável: retornaRAs[] = vetor que contém listadas alfabeticamente todas as regiões administrativas.
    - Função: consultarPorId($id)
      * Variável: id = será usada pra pesquisar um id específico;
      * Variável: sql = usada pra pesquisar um id específico da tabela regiao\_administrativa;
      * Variável: resultado
      * Variável: registro = acessa um registro da tabela ;
      * Variável: dadosRA = contém os dados da regiao\_administrativa acessados pelo id que foi passado como parâmetro;
      * Variável: retornaRAs[]
    - Função: consultarPorNome($nome)
      * Variável: nome = será usada pra pesquisar um registro de um nome específico;
      * Variável: sql = usada pra pesquisar um nome específico da tabela regiao\_administrativa;
      * Variável: resultado
      * Variável: registro = acessa um registro da tabela ;
      * Variável: dadosRA = contém os dados da regiao\_administrativa acessados pelo nome que foi passado como parâmetro;
      * Variável: retornaRAs[]
    - Função: contarRegistrosRA()
      * Variável: sql = seleciona a função do banco de dados de contar quantas regiões administrativas existem pela conta do número de ids distintos da tabela regiao\_administrativa;
      * Variável: resultado
      * Variável: registro = recebe a quantidade de regiões administrativas existentes.
  + Classe: Conexao.php
  + Classe: ConexaoTeste.php
  + Classe: CrimeDAO.php
  + Classe: NaturezaDAO.php
  + Classe: RegiaoAdministrativaDAO.php
    - * Variável: conexao = variável usada pra instanciar um objeto da classe Conexao pra estabelecer conexão com o banco de dados.
    - Função: \_\_constructTeste()
      * Variável: conexao
    - Função: \_\_constructTeste()
      * Variável: conexao
    - Função: listarTodas()
      * Variável: sql = usada pra acessar a tabela regiao\_administrativa
      * Variável: resultado = recebe o resultado, se foi bem sucedida, da seleção da tabela regiao\_administrativa
  + Classe: TempoDAO.php
  + Classe: persistence.php
* **Pacote: test**
* **Pasta: controller**
  + Classe: CategoriaControllerTeste.php
  + Classe: CrimeControllerTeste.php
  + Classe: NaturezaControllerTeste.php
  + Classe: RegiaoAdministrativaControllerTeste.php
  + Classe: TempoControllerTeste.php
* **Pasta: model**
  + Classe: CategoriaTeste.php
    - Função: setUp
      * Variável: categoria = nova instância
    - Função: testSetIdCategoria
      * Variável: categoria
    - Função: testExceptionSetIdCategoria
      * Variável: categoria
    - Função: testSetNomeCategoria
      * Variável: categoria
    - Função: testExceptionSetNomeCategoria
      * Variável: categoria
    - Função: testConstructOverLoad
      * Variável: categoria
  + Classe: CrimeTeste.php
    - Função: setUp
      * Variável: crime = instancia novo crime
    - Função: testeAtributoIdCrime
      * Variável: crime
    - Função: testExceptionSetIdCrime
      * Variável: crime
    - Função: testeAtributoQuantidade
      * Variável: crime
    - Função: testeIdTempo
      * Variável: crime
    - Função: testeIdNatureza
      * Variável: crime
    - Função: testeConstructOverLoad
      * Variável: crime
    - Função: testeIdRA
      * Variável: crime
  + Classe: NaturezaTeste.php
  + Classe: RegiaoAdministrativaTeste.php
  + Classe: TempoTeste.php
* **Pasta: persistence**
  + Classe: Conexao.php
    - * Variável: banco = usada para criar conexão com o banco de tipo\_banco;
      * Variável: tipo\_banco = contém o tipo de banco, no caso é mysql;
      * Variável: servidor = contém o nome do servidor, no caso é localhost;
      * Variável: usuario = contém o nome do usuário que irá acessar o banco;
      * Variável: senha = contém a senha do usuário que irá acessar o banco;
  + Classe: persistence.php
    - * Variável: db = cria uma instancia do banco de dados.
    - Função: getInstance()
* **Pasta: util**
  + Classe:
* **Pacote: util**
  + Classe: Parse.php
  + Classe: RunParse.php
* **Pacote: view**
  + Classe: totalra.php
    - * Variável: crimeVW = usada pra instanciar a view de crime
      * Variável: tempoVW = usada pra instanciar a view de crime
  + Classe: viewController.php
    - * Variável: pagina = usada para mudar de página (página ano, página tipo, página index)
  + Classe: router.php
    - * Variável: pagina = usada para mudar de página (página ano, página tipo, página total ra, página crime por categoria, página initial)
* **Pacote: views**
  + Classe: