**2。API概览：**  
class Reflection { }  
interface Reflector { }  
class ReflectionException extends Exception { }  
class ReflectionFunction implements Reflector { }  
class ReflectionParameter implements Reflector { }  
class ReflectionMethod extends ReflectionFunction { }  
class ReflectionClass implements Reflector { }  
class ReflectionObject extends ReflectionClass { }  
class ReflectionProperty implements Reflector { }  
class ReflectionExtension implements Reflector { }

**3。详细说明：（例子详见php手册）**①Reflection类  
<?php  
class Reflection  
{  
public static mixed export(Reflector r [,bool return])  
//导出一个类或方法的详细信息  
public static array getModifierNames(int modifiers)  
//取得修饰符的名字  
}  
?>

②ReflectionException类

该类继承标准类，没特殊方法和属性。

③ReflectionFunction类  
<?php  
class ReflectionFunction implements Reflector  
{  
final private \_\_clone()  
public object \_\_construct(string name)  
public string \_\_toString()  
public static string export()  
//导出该函数的详细信息  
public string getName()  
//取得函数名  
public bool isInternal()  
//测试是否为系统内部函数  
public bool isUserDefined()  
//测试是否为用户自定义函数  
public string getFileName()  
//取得文件名，包括路径名  
public int getStartLine()  
//取得定义函数的起始行  
public int getEndLine()  
//取得定义函数的结束行  
public string getDocComment()  
//取得函数的注释  
public array getStaticVariables()  
//取得静态变量  
public mixed invoke(mixed\* args)  
//调用该函数，通过参数列表传参数  
public mixed invokeArgs(array args)  
//调用该函数，通过数组传参数  
public bool returnsReference()  
//测试该函数是否返回引用  
public ReflectionParameter[] getParameters()  
//取得该方法所需的参数，返回值为对象数组  
public int getNumberOfParameters()  
//取得该方法所需的参数个数  
public int getNumberOfRequiredParameters()  
//取得该方法所需的参数个数  
}  
?>

④ReflectionParameter类：  
<?php  
class ReflectionParameter implements Reflector  
{  
final private \_\_clone()  
public object \_\_construct(string name)  
public string \_\_toString()  
public static string export()  
//导出该参数的详细信息  
public string getName()  
//取得参数名  
public bool isPassedByReference()  
//测试该参数是否通过引用传递参数  
public ReflectionClass getClass()  
//若该参数为对象，返回该对象的类名  
public bool isArray()  
//测试该参数是否为数组类型  
public bool allowsNull()  
//测试该参数是否允许为空  
public bool isOptional()  
//测试该参数是否为可选的，当有默认参数时可选  
public bool isDefaultValueAvailable()  
//测试该参数是否为默认参数  
public mixed getDefaultValue()  
//取得该参数的默认值  
}  
?>

⑤ReflectionClass类：  
<?php  
class ReflectionClass implements Reflector  
{  
final private \_\_clone()  
public object \_\_construct(string name)  
public string \_\_toString()  
public static string export()  
//导出该类的详细信息  
public string getName()  
//取得类名或接口名  
public bool isInternal()  
//测试该类是否为系统内部类  
public bool isUserDefined()  
//测试该类是否为用户自定义类  
public bool isInstantiable()  
//测试该类是否被实例化过  
public bool hasConstant(string name)  
//测试该类是否有特定的常量  
public bool hasMethod(string name)  
//测试该类是否有特定的方法  
public bool hasProperty(string name)  
//测试该类是否有特定的属性  
public string getFileName()  
//取得定义该类的文件名，包括路径名  
public int getStartLine()  
//取得定义该类的开始行  
public int getEndLine()  
//取得定义该类的结束行  
public string getDocComment()  
//取得该类的注释  
public ReflectionMethod getConstructor()  
//取得该类的构造函数信息  
public ReflectionMethod getMethod(string name)  
//取得该类的某个特定的方法信息  
public ReflectionMethod[] getMethods()  
//取得该类的所有的方法信息  
public ReflectionProperty getProperty(string name)  
//取得某个特定的属性信息  
public ReflectionProperty[] getProperties()  
//取得该类的所有属性信息  
public array getConstants()  
//取得该类所有常量信息  
public mixed getConstant(string name)  
//取得该类特定常量信息  
public ReflectionClass[] getInterfaces()  
//取得接口类信息  
public bool isInterface()  
//测试该类是否为接口  
public bool isAbstract()  
//测试该类是否为抽象类  
public bool isFinal()  
//测试该类是否声明为final  
public int getModifiers()  
//取得该类的修饰符，返回值类型可能是个资源类型  
//通过Reflection::getModifierNames($class->getModifiers())进一步读取  
public bool isInstance(stdclass object)  
//测试传入的对象是否为该类的一个实例  
public stdclass newInstance(mixed\* args)  
//创建该类实例  
public ReflectionClass getParentClass()  
//取得父类  
public bool isSubclassOf(ReflectionClass class)  
//测试传入的类是否为该类的父类  
public array getStaticProperties()  
//取得该类的所有静态属性  
public mixed getStaticPropertyValue(string name [, mixed default])  
//取得该类的静态属性值，若private，则不可访问  
public void setStaticPropertyValue(string name, mixed value)  
//设置该类的静态属性值，若private，则不可访问，有悖封装原则  
public array getDefaultProperties()  
//取得该类的属性信息，不含静态属性  
public bool isIterateable()  
public bool implementsInterface(string name)  
//测试是否实现了某个特定接口  
public ReflectionExtension getExtension()  
public string getExtensionName()  
}  
?>

⑥ReflectionMethod类：  
<?php  
class ReflectionMethod extends ReflectionFunction  
{  
public \_\_construct(mixed class, string name)  
public string \_\_toString()  
public static string export()  
//导出该方法的信息  
public mixed invoke(stdclass object, mixed\* args)  
//调用该方法  
public mixed invokeArgs(stdclass object, array args)  
//调用该方法，传多参数  
public bool isFinal()  
//测试该方法是否为final  
public bool isAbstract()  
//测试该方法是否为abstract  
public bool isPublic()  
//测试该方法是否为public  
public bool isPrivate()  
//测试该方法是否为private  
public bool isProtected()  
//测试该方法是否为protected  
public bool isStatic()  
//测试该方法是否为static  
public bool isConstructor()  
//测试该方法是否为构造函数  
public bool isDestructor()  
//测试该方法是否为析构函数  
public int getModifiers()  
//取得该方法的修饰符  
public ReflectionClass getDeclaringClass()  
//取得该方法所属的类  
// Inherited from ReflectionFunction  
final private \_\_clone()  
public string getName()  
public bool isInternal()  
public bool isUserDefined()  
public string getFileName()  
public int getStartLine()  
public int getEndLine()  
public string getDocComment()  
public array getStaticVariables()  
public bool returnsReference()  
public ReflectionParameter[] getParameters()  
public int getNumberOfParameters()  
public int getNumberOfRequiredParameters()  
}  
?>

⑦ReflectionProperty类：  
<?php  
class ReflectionProperty implements Reflector  
{  
final private \_\_clone()  
public \_\_construct(mixed class, string name)  
public string \_\_toString()  
public static string export()  
//导出该属性的详细信息  
public string getName()  
//取得该属性名  
public bool isPublic()  
//测试该属性名是否为public  
public bool isPrivate()  
//测试该属性名是否为private  
public bool isProtected()  
//测试该属性名是否为protected  
public bool isStatic()  
//测试该属性名是否为static  
public bool isDefault()  
public int getModifiers()  
//取得修饰符  
public mixed getValue(stdclass object)  
//取得该属性值  
public void setValue(stdclass object, mixed value)  
//设置该属性值  
public ReflectionClass getDeclaringClass()  
//取得定义该属性的类  
public string getDocComment()  
//取得该属性的注释  
}  
?>

⑧ReflectionExtension类  
<?php  
class ReflectionExtension implements Reflector {  
final private \_\_clone()  
public \_\_construct(string name)  
public string \_\_toString()

public static string export()  
//导出该扩展的所有信息  
public string getName()  
//取得该扩展的名字  
public string getVersion()  
//取得该扩展的版本  
public ReflectionFunction[] getFunctions()  
//取得该扩展的所有函数  
public array getConstants()  
//取得该扩展的所有常量  
public array getINIEntries()  
//取得与该扩展相关的，在php.ini中的指令信息  
public ReflectionClass[] getClasses()  
public array getClassNames()  
}  
?>