2022-07-12

1.linux umask

查看umask:umask

设置umask:umask 0077

文件权限：777-umask

Example:将文件权限设置为对owner读写

umask 0077

touch aa

ls -l aa

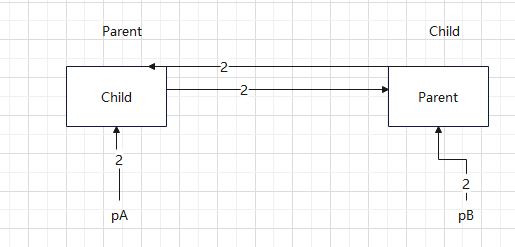
则可看出文件权限



2.shared\_ptr

new一个对象，shared\_ptr引用计数器会加1，将一个shared\_ptr拷贝到另一个shared\_ptr对象，引用计数器会加1，但使用引用传递，不会导致引用计数器加1。

当存在循环依赖关系，



造成内存泄漏。

3.weak\_ptr

只是指向内存，通过Lock定位到相应内存，获得shared\_ptr，shared\_ptr的reset是释放指针，引用计数器减一。

2022-07-13

1.android timePicker

时间控件，可以获取时间：TimePicker.getCurrentHour()、TimePicker.getCurrentMinute()

2.boost::base\_from\_member

当一个类继承boost::base\_from\_member，便可通过::member访问其对象。如下所示：

#include <iostream>

#include <sstream>

#include "boost/noncopyable.hpp"

#include "boost/utility/base\_from\_member.hpp"

class task\_processor : boost::noncopyable {

    std::ostream& log\_;

protected:

    virtual void do\_process() = 0;

public:

    explicit task\_processor(std::ostream& log) : log\_(log) {}

    void process() {

        log\_ << "starting process..." << std::endl;

        do\_process();

    }

};

class task\_processor\_fixed : boost::base\_from\_member<std::ostringstream>

                           , public task\_processor {

    typedef boost::base\_from\_member<std::ostringstream> logger\_t;

    virtual void do\_process() {

        logger\_t::member << "fix processor processing..." << std::endl;

    }

public:

    task\_processor\_fixed()

            : logger\_t()

            , task\_processor(logger\_t::member)

    {}

};

int main() {

    task\_processor\_fixed tf;

    tf.process();

    return 0;

}

3.情商

偏见是个体早期获得的一种情绪经验，偏见的情绪在童年时形成，但是支持偏见的信念是后来才出现的，偏见反应很难完全消除，可以改变的是人们对待偏见的态度！

一个工作团队内的成员，其优势和视角各有千秋，如果他们能够协调一致，相对于同一批人各自独立工作，可能会实现更出色，更有创造性，更有效的解决方案！

4.Java

abstract既可以定义类，也可以定义方法，定义类时代表该类是抽象类，没有具体实现，需要在派生类实现abstract方法。

JAVA类默认访问权限是public，具体demo：

**abstract** **class** Example0507 {

**int** x;

**abstract** **int** m1();

**abstract** **int** m2();

}

**abstract** **class** Example0507\_1 **extends** Example0507 {

**int** y;

**int** m1() {

**return** x + y;

}

}

**class** Example0507\_2 **extends** Example0507\_1 {

**int** z;

**int** m2() {

**return** x + y + z;

}

}

**public** **class** Hello {

**public** **static** **void** main(String[] args) {

Example0507\_2 mm = **new** Example0507\_2();

mm.x = 10;

mm.y = 20;

mm.z = 30;

System.***out***.print(mm.m2());

}

}