

KBSW180120 Win32-CompositeMap

composite_map_demo, .stcmcomposite map

-
-
-
-
-

- - Visual Studio 2010 SP1
 - Slamware Windows SDK:[Slamware Windows SDK](#)
 - RoboStudio():[Robostudio installer](#)
 - Sample Code:



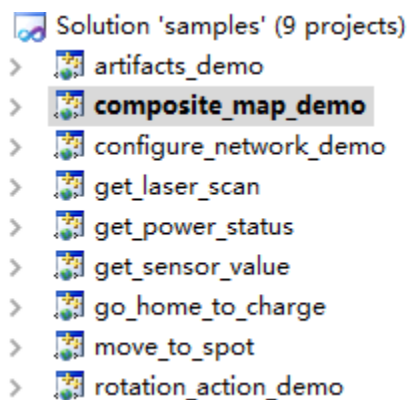
Visual Studio

Visual Studio 2010SP1.Net FrameworkSP1

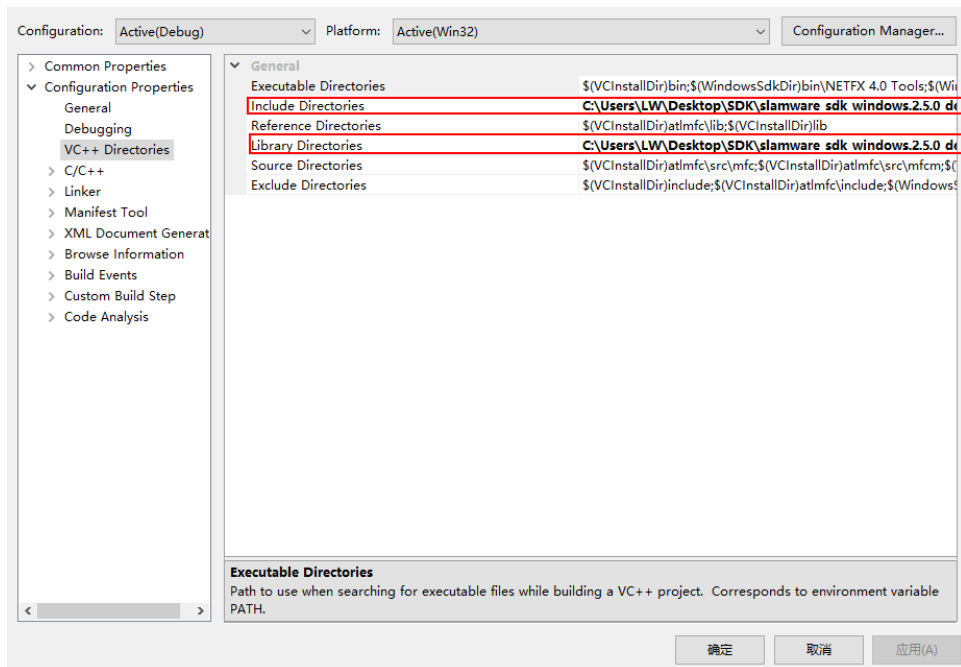
- - Slamware SDP mini
 - Slamware SDP
 - Slamware Slamware
 - Zeus/Apollo

[Win32-](#)

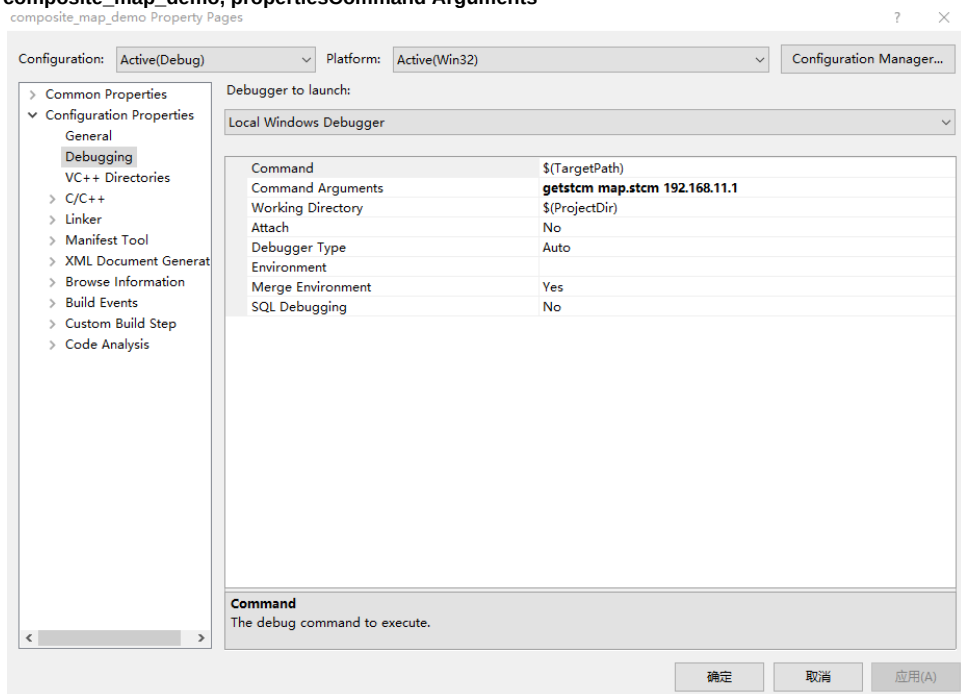
1. samplescomposite_map_demo, StartUp project



2. composite_map_demo, Slamware SDK includelib



3. composite_map_demo, propertiesCommand Arguments



composite_map_demo [OPTS] [filename] <SDP IP Address>
 SDP IP Address The ip address string of the SLAMWARE SDP
 getstcm filename download compositeMap
 setstcm filename upload compositeMap
 -h Show this message

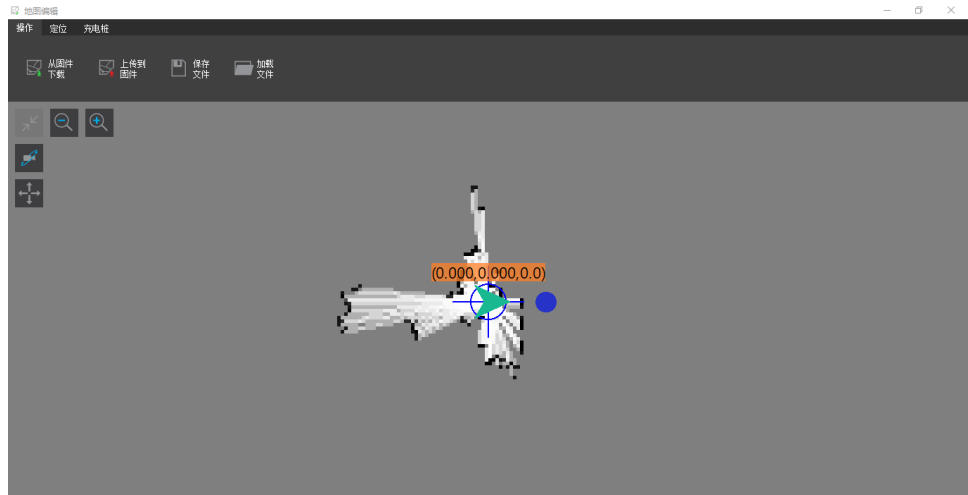
4. F5

- **composite_map_demo** getstcm map.stcm 192.168.11.1 (slamwarecomposite map, map.stcm)

```

Connecting to 10.0.0.1: 10.0.0.1...
Nmap Scan Report
Host: 10.0.0.1
OS: Linux 3.2.0-4-amd64 (Debian 7.0)
Nmap Version: 7.80
Scan Date: 2023-12-12 10:17:12
Scan Time: 0.00s
Hosts: 10.0.0.1
Ports: 22/tcp, 80/tcp, 443/tcp, 8080/tcp, 8443/tcp, 9090/tcp, 9443/tcp, 9543/tcp, 9643/tcp, 9743/tcp, 9843/tcp, 9943/tcp, 10000/tcp, 10001/tcp, 10002/tcp, 10003/tcp, 10004/tcp, 10005/tcp, 10006/tcp, 10007/tcp, 10008/tcp, 10009/tcp, 10010/tcp, 10011/tcp, 10012/tcp, 10013/tcp, 10014/tcp, 10015/tcp, 10016/tcp, 10017/tcp, 10018/tcp, 10019/tcp, 10020/tcp, 10021/tcp, 10022/tcp, 10023/tcp, 10024/tcp, 10025/tcp, 10026/tcp, 10027/tcp, 10028/tcp, 10029/tcp, 10030/tcp, 10031/tcp, 10032/tcp, 10033/tcp, 10034/tcp, 10035/tcp, 10036/tcp, 10037/tcp, 10038/tcp, 10039/tcp, 10040/tcp, 10041/tcp, 10042/tcp, 10043/tcp, 10044/tcp, 10045/tcp, 10046/tcp, 10047/tcp, 10048/tcp, 10049/tcp, 10050/tcp, 10051/tcp, 10052/tcp, 10053/tcp, 10054/tcp, 10055/tcp, 10056/tcp, 10057/tcp, 10058/tcp, 10059/tcp, 10060/tcp, 10061/tcp, 10062/tcp, 10063/tcp, 10064/tcp, 10065/tcp, 10066/tcp, 10067/tcp, 10068/tcp, 10069/tcp, 10070/tcp, 10071/tcp, 10072/tcp, 10073/tcp, 10074/tcp, 10075/tcp, 10076/tcp, 10077/tcp, 10078/tcp, 10079/tcp, 10080/tcp, 10081/tcp, 10082/tcp, 10083/tcp, 10084/tcp, 10085/tcp, 10086/tcp, 10087/tcp, 10088/tcp, 10089/tcp, 10090/tcp, 10091/tcp, 10092/tcp, 10093/tcp, 10094/tcp, 10095/tcp, 10096/tcp, 10097/tcp, 10098/tcp, 10099/tcp, 10100/tcp, 10101/tcp, 10102/tcp, 10103/tcp, 10104/tcp, 10105/tcp, 10106/tcp, 10107/tcp, 10108/tcp, 10109/tcp, 10110/tcp, 10111/tcp, 10112/tcp, 10113/tcp, 10114/tcp, 10115/tcp, 10116/tcp, 10117/tcp, 10118/tcp, 10119/tcp, 10120/tcp, 10121/tcp, 10122/tcp, 10123/tcp, 10124/tcp, 10125/tcp, 10126/tcp, 10127/tcp, 10128/tcp, 10129/tcp, 10130/tcp, 10131/tcp, 10132/tcp, 10133/tcp, 10134/tcp, 10135/tcp, 10136/tcp, 10137/tcp, 10138/tcp, 10139/tcp, 10140/tcp, 10141/tcp, 10142/tcp, 10143/tcp, 10144/tcp, 10145/tcp, 10146/tcp, 10147/tcp, 10148/tcp, 10149/tcp, 10150/tcp, 10151/tcp, 10152/tcp, 10153/tcp, 10154/tcp, 10155/tcp, 10156/tcp, 10157/tcp, 10158/tcp, 10159/tcp, 10160/tcp, 10161/tcp, 10162/tcp, 10163/tcp, 10164/tcp, 10165/tcp, 10166/tcp, 10167/tcp, 10168/tcp, 10169/tcp, 10170/tcp, 10171/tcp, 10172/tcp, 10173/tcp, 10174/tcp, 10175/tcp, 10176/tcp, 10177/tcp, 10178/tcp, 10179/tcp, 10180/tcp, 10181/tcp, 10182/tcp, 10183/tcp, 10184/tcp, 10185/tcp, 10186/tcp, 10187/tcp, 10188/tcp, 10189/tcp, 10190/tcp, 10191/tcp, 10192/tcp, 10193/tcp, 10194/tcp, 10195/tcp, 10196/tcp, 10197/tcp, 10198/tcp, 10199/tcp, 10200/tcp, 10201/tcp, 10202/tcp, 10203/tcp, 10204/tcp, 10205/tcp, 10206/tcp, 10207/tcp, 10208/tcp, 10209/tcp, 10210/tcp, 10211/tcp, 10212/tcp, 10213/tcp, 10214/tcp, 10215/tcp, 10216/tcp, 10217/tcp, 10218/tcp, 10219/tcp, 10220/tcp, 10221/tcp, 10222/tcp, 10223/tcp, 10224/tcp, 10225/tcp, 10226/tcp, 10227/tcp, 10228/tcp, 10229/tcp, 10230/tcp, 10231/tcp, 10232/tcp, 10233/tcp, 10234/tcp, 10235/tcp, 10236/tcp, 10237/tcp, 10238/tcp, 10239/tcp, 10240/tcp, 10241/tcp, 10242/tcp, 10243/tcp, 10244/tcp, 10245/tcp, 10246/tcp, 10247/tcp, 10248/tcp, 10249/tcp, 10250/tcp, 10251/tcp, 10252/tcp, 10253/tcp, 10254/tcp, 10255/tcp, 10256/tcp, 10257/tcp, 10258/tcp, 10259/tcp, 10260/tcp, 10261/tcp, 10262/tcp, 10263/tcp, 10264/tcp, 10265/tcp, 10266/tcp, 10267/tcp, 10268/tcp, 10269/tcp, 10270/tcp, 10271/tcp, 10272/tcp, 10273/tcp, 10274/tcp, 10275/tcp, 10276/tcp, 10277/tcp, 10278/tcp, 10279/tcp, 10280/tcp, 10281/tcp, 10282/tcp, 10283/tcp, 10284/tcp, 10285/tcp, 10286/tcp, 10287/tcp, 10288/tcp, 10289/tcp, 10290/tcp, 10291/tcp, 10292/tcp, 10293/tcp, 10294/tcp, 10295/tcp, 10296/tcp, 10297/tcp, 10298/tcp, 10299/tcp, 10300/tcp, 10301/tcp, 10302/tcp, 10303/tcp, 10304/tcp, 10305/tcp, 10306/tcp, 10307/tcp, 10308/tcp, 10309/tcp, 10310/tcp, 10311/tcp, 10312/tcp, 10313/tcp, 10314/tcp, 10315/tcp, 10316/tcp, 10317/tcp, 10318/tcp, 10319/tcp, 10320/tcp, 10321/tcp, 10322/tcp, 10323/tcp, 10324/tcp, 10325/tcp, 10326/tcp, 10327/tcp, 10328/tcp, 10329/tcp, 10330/tcp, 10331/tcp, 10332/tcp, 10333/tcp, 10334/tcp, 10335/tcp, 10336/tcp, 10337/tcp, 10338/tcp, 10339/tcp, 10340/tcp, 10341/tcp, 10342/tcp, 10343/tcp, 10344/tcp, 10345/tcp, 10346/tcp, 10347/tcp, 10348/tcp, 10349/tcp, 10350/tcp, 10351/tcp, 10352/tcp, 10353/tcp, 10354/tcp, 10355/tcp, 10356/tcp, 10357/tcp, 10358/tcp, 10359/tcp, 10360/tcp, 10361/tcp, 10362/tcp, 10363/tcp, 10364/tcp, 10365/tcp, 10366/tcp, 10367/tcp, 10368/tcp, 10369/tcp, 10370/tcp, 10371/tcp, 10372/tcp, 10373/tcp, 10374/tcp, 10375/tcp, 10376/tcp, 10377/tcp, 10378/tcp, 10379/tcp, 10380/tcp, 10381/tcp, 10382/tcp, 10383/tcp, 10384/tcp, 10385/tcp, 10386/tcp, 10387/tcp, 10388/tcp, 10389/tcp, 10390/tcp, 10391/tcp, 10392/tcp, 10393/tcp, 10394/tcp, 10395/tcp, 10396/tcp, 10397/tcp, 10398/tcp, 10399/tcp, 10400/tcp, 10401/tcp, 10402/tcp, 10403/tcp, 10404/tcp, 10405/tcp, 10406/tcp, 10407/tcp, 10408/tcp, 10409/tcp, 10410/tcp, 10411/tcp, 10412/tcp, 10413/tcp, 10414/tcp, 10415/tcp, 10416/tcp, 10417/tcp, 10418/tcp, 10419/tcp, 10420/tcp, 10421/tcp, 10422/tcp, 10423/tcp, 10424/tcp, 10425/tcp, 10426/tcp, 10427/tcp, 10428/tcp, 10429/tcp, 10430/tcp, 10431/tcp, 10432/tcp, 10433/tcp, 10434/tcp, 10435/tcp, 10436/tcp, 10437/tcp, 10438/tcp, 10439/tcp, 10440/tcp, 10441/tcp, 10442/tcp, 10443/tcp, 10444/tcp, 10445/tcp, 10446/tcp, 10447/tcp, 10448/tcp, 10449/tcp, 10450/tcp, 10451/tcp, 10452/tcp, 10453/tcp, 10454/tcp, 10455/tcp, 10456/tcp, 10457/tcp, 10458/tcp, 10459/tcp, 10460/tcp, 10461/tcp, 10462/tcp, 10463/tcp, 10464/tcp, 10465/tcp, 10466/tcp, 10467/tcp, 10468/tcp, 10469/tcp, 10470/tcp, 10471/tcp, 10472/tcp, 10473/tcp, 10474/tcp, 10475/tcp, 10476/tcp, 10477/tcp, 10478/tcp, 10479/tcp, 10480/tcp, 10481/tcp, 10482/tcp, 10483/tcp, 10484/tcp, 10485/tcp, 10
```

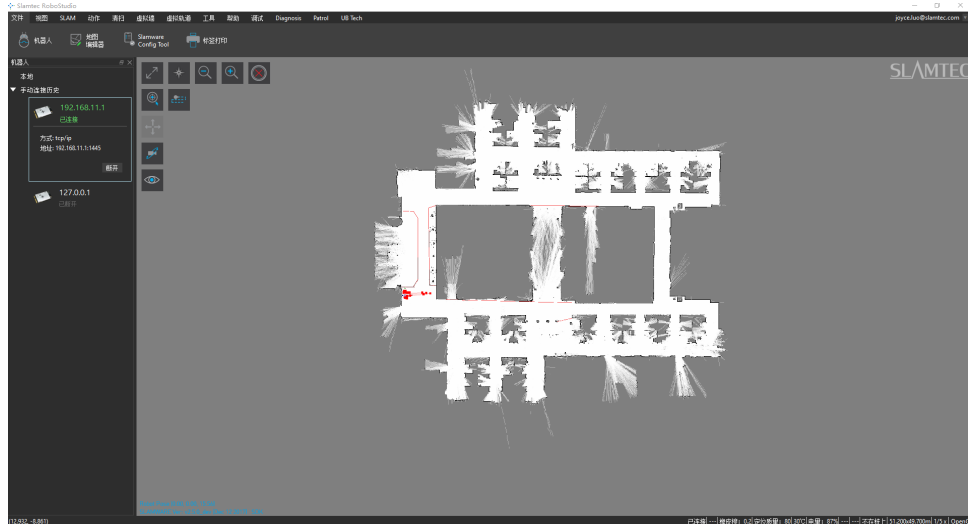
Robostudio



- **composite_map_demo setstcm map.stcm 192.168.11.1(map.stcmlamware)**

[illegible]

Robostudio



- slamwarecomposite map

composite map

```
bool StcmMapWriter(const std::string file_name, SlamwareCorePlatform platform) {
    CompositeMap composite_map = platform.getCompositeMap();
    CompositeMapWriter composite_map_writer;
    std::string error_message;
    bool result = composite_map_writer.saveFile(error_message, file_name, composite_map);
    return result;
}
```

- composite mapslamware

composite map

```
bool StcmMapReader(const std::string file_path, rpos::core::Pose pose, SlamwareCorePlatform platform) {
    CompositeMapReader composite_map_reader;
    std::string error_message;
    boost::shared_ptr<CompositeMap> composite_map(composite_map_reader.loadFile(error_message, file_path));
    if (composite_map) {
        platform.setCompositeMap((*composite_map), pose);
        return true;
    }
    return false;
}
```

- composite map

map layer

```
CompositeMapReader composite_map_reader;
std::string error_message;
boost::shared_ptr<CompositeMap> composite_map(composite_map_reader.loadFile(error_message, file_path));
if (composite_map) {
    for (auto it = composite_map->maps().begin(); it != composite_map->maps().end(); ++it) {
        auto layer = *it;
        std::string usage = layer->getUsage();
        std::string type = layer->getType();
        std::cout << "Layer Usage : " << usage << std::endl;
        //get grid map layer
        if (type == GridMapLayer::Type) {
            auto grid_map = boost::dynamic_pointer_cast<GridMapLayer>(layer);
            std::cout << "Map Position : (" << grid_map->getOrigin().x() << " , " <<
                grid_map->getOrigin().y() << ")" <<std::endl;
            std::cout << "Map Resolution : (" << grid_map->getResolution().x() <<
                " , " << grid_map->getResolution().y() << ")" <<std::endl;
            std::cout << "Map Dimension: (" << grid_map->getDimension().x() <<
                " , " << grid_map->getDimension().y() << ")" <<std::endl;
            std::cout << "Map Data:" << std::endl;
            for (auto it = grid_map->mapData().begin(); it != grid_map->mapData().end();
++it) {
                std::cout << (int)*it << " " ;
            }
            std::cout << std::endl << std::endl;
        }
        //get line map layer
        else if (type == LineMapLayer::Type) {
            auto line_map = boost::dynamic_pointer_cast<LineMapLayer>(layer);
            for (auto it = line_map->lines().begin(); it != line_map->lines().end();
++it) {
                auto line = it->second;
                std::cout << "start: (" << line.start.x() << " , " << line.start.y()
<< ")" << std::endl;
                std::cout << "end: (" << line.end.x() << " , " << line.end.y() <<
                ")" << std::endl;
            }
        }
    }
}
```

```

        }
        std::cout << std::endl;
    }

    //get pose map layer
    else if (type == PoseMapLayer::Type) {
        auto pose_map = boost::dynamic_pointer_cast<PoseMapLayer>(layer);
        for (auto it = pose_map->poses().begin(); it != pose_map->poses().end();
++it) {
            auto pos = it->second;
            std::cout << "Position : (" << pos.pose.x() << " , " << pos.pose.y()
<< ")" << std::endl;
        }
        std::cout << std::endl;
    }
    else if (type == PointsMapLayer::Type) {
        //TODO: get Points map layer
        std::cout << std::endl;
    }
    else {
        //TODO: get unknown map layer
        std::cout << std::endl;
    }
}
}

```