Ford Motor (China) Company

SDL 中 SetDisplayLayout 流程分析

Version <1.0>

免责申明

本文档中的内容仅供参考。福特汽车中国对本服务内容的错误或遗漏概不负责。在任何情况下，福特汽车中国均不对因使用本文档而产生或与之相关的任何特殊，直接，间接，间接或偶然的损害赔偿或任何损害负责，无论是在合同，疏忽，其他侵权行为中服务或服务的内容。福特汽车中国保留随时对本文档内容进行补充，删除或修改的权利，恕不另行通知。

Disclaimer

The contents contained in this document are for general information purposes only. Ford Motor China assumes no responsibility for errors or omissions in the contents on the Service.
In no event shall Ford Motor China be liable for any special, direct, indirect, consequential, or incidental damages or any damages whatsoever, whether in an action of contract, negligence or other tort, arising out of or in connection with the use of the Document or the contents of the Document.
Ford Motor China reserves the right to make additions, deletions, or modification to the contents on the Service at any time without prior notice.

编制	Beyondsoft	日期	2017-11-20	版权	署名-相同方式共享 4.0 国际
审核	Ford	日期	2017-11-20	管理	Ford

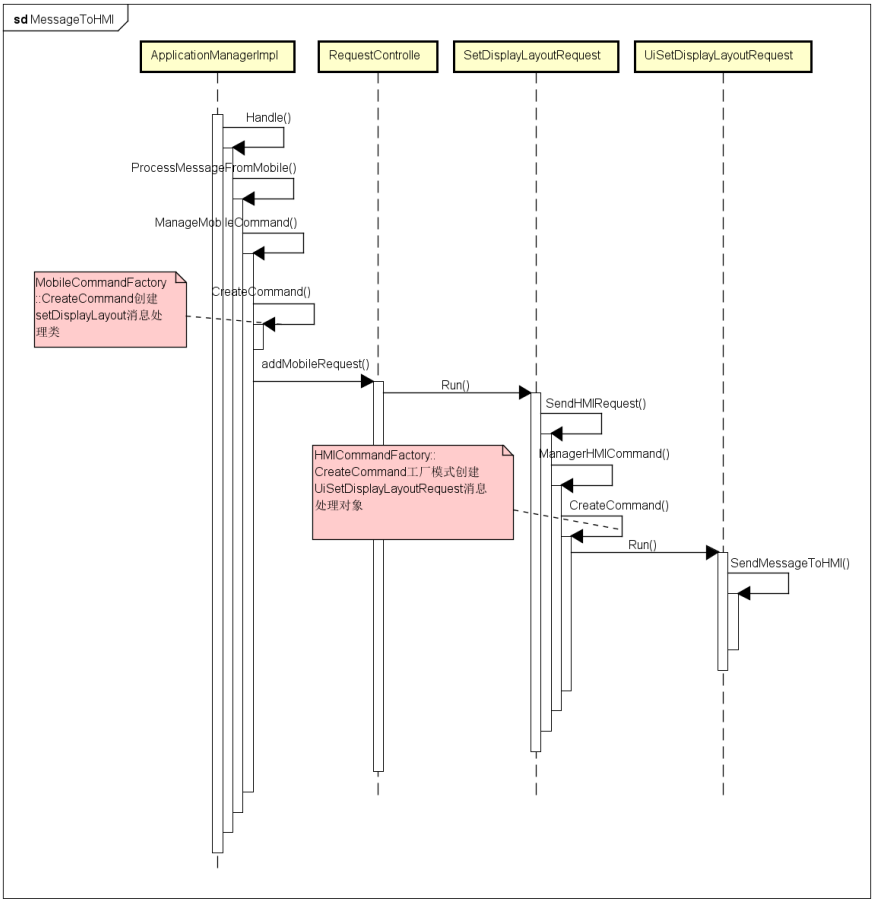
Project Name: sdl_core	Version: <1.0>
SDL 中 SetDisplayLayout 协议流程分析	Date: <20/11/2017>
<document identifier>	

修改历史

版本	日期	说明
1.0.0	2017-11-20	初版，完成文档架构，填写文档内容
1.0.1	2017-11-24	增加流程图

目录

1 消息发送流程



2 MessageFromMobile 消息处理

2.1 Handle

```
void ApplicationManagerImpl::Handle(const impl::MessageFromMobile message) {
  LOG4CXX_AUTO_TRACE(logger_);

  if (!message) {
    LOG4CXX_ERROR(logger_, "Null-pointer message received.");
    return;
  }
  sync_primitives::AutoLock lock(stopping_flag_lock_);
  if (is_stopping_) {
    LOG4CXX_INFO(logger_, "Application manager is stopping");
    return;
  }
  ProcessMessageFromMobile(message);
}
```

2.2 ProcessMessageFromMobile

```
void ApplicationManagerImpl::ProcessMessageFromMobile(
    const utils::SharedPtr<Message> message) {
  LOG4CXX_AUTO_TRACE(logger_);
#ifdef TELEMETRY_MONITOR
  AMTelemetryObserver::MessageMetricSharedPtr metric(
      new AMTelemetryObserver::MessageMetric());
  metric->begin = date_time::DateTime::getCurrentTime();
#endif  // TELEMETRY_MONITOR
  smart_objects::SmartObjectSPtr so_from_mobile(new smart_objects::SmartObject);

  if (!so_from_mobile) {
    LOG4CXX_ERROR(logger_, "Null pointer");
    return;
  }

  if (!ConvertMessageToSO(*message, *so_from_mobile)) {
    LOG4CXX_ERROR(logger_, "Cannot create smart object from message");
    return;
  }
#ifdef TELEMETRY_MONITOR
  metric->message = so_from_mobile;
#endif  // TELEMETRY_MONITOR

  if (!ManageMobileCommand(so_from_mobile, commands::Command::ORIGIN_MOBILE)) {
    LOG4CXX_ERROR(logger_, "Received command didn't run successfully");
  }
#ifdef TELEMETRY_MONITOR
  metric->end = date_time::DateTime::getCurrentTime();
  if (metric_observer_) {
    metric_observer_->OnMessage(metric);
  }
#endif  // TELEMETRY_MONITOR
}
```

2.3 ManageMobileCommand

```
bool ApplicationManagerImpl::ManageMobileCommand(
    const commands::MessageSharedPtr message,
    commands::Command::CommandOrigin origin) {
  LOG4CXX_AUTO_TRACE(logger_);

  if (!message) {
    LOG4CXX_WARN(logger_, "Null-pointer message received.");
    return false;
  }

  if (IsLowVoltage()) {
    LOG4CXX_WARN(logger_, "Low Voltage is active");
    return false;
  }
#ifdef DEBUG
  MessageHelper::PrintSmartObject(*message);
#endif

  LOG4CXX_DEBUG(logger_, "T                                              ;
  utils::SharedPtr<commands::Command> command(
      MobileCommandFactory::CreateCommand(message, origin));
```

MobileCommandFactory::CreateCommand 工厂模式根据 RPC 定义的 ID 和消息来源创建对应 RPC 消息处理对象

2.4 RequestController

setDisplayLayout 的请求类型会将 command 加入到请求控制器（RequestController）中进行处理

```
if (message_type == mobile_apis::messageType::request) {
  // commands will be launched from requesr_ctrl
  mobile_apis::HMILevel::eType app_hmi_level =
      mobile_apis::HMILevel::INVALID_ENUM;
  if (app) {
    app_hmi_level = app->hmi_level();
  }

  // commands will be launched from request_ctrl

  const request_controller::RequestController::TResult result =
      request_ctrl_.addMobileRequest(command, app_hmi_level);
```

3 RequestController

setDisplayLayout 请求被加入到 RequestController 中，通过 request_ptr->Run();调用内部处理模块

```
void RequestController::Worker::threadMain() {
  LOG4CXX_AUTO_TRACE(logger_);
  AutoLock auto_lock(thread_lock_);
  while (!stop_flag_) {
    // Try to pick a request
    AutoLock auto_lock(request_controller_->mobile_request_list_lock_);

    while ((request_controller_->pool_state_ != TPoolState::STOPPED) &&
           (request_controller_->mobile_request_list_.empty())) {
      // Wait until there is a task in the queue
      // Unlock mutex while wait, then lock it back when signaled
      LOG4CXX_INFO(logger_, "Unlocking and waiting");
      request_controller_->cond_var_.Wait(auto_lock);
      LOG4CXX_INFO(logger_, "Signaled and locking");
    }

    // If the thread was shutdown, return from here
    if (request_controller_->pool_state_ == TPoolState::STOPPED) {
      break;
    }

    if (request_controller_->mobile_request_list_.empty()) {
      LOG4CXX_WARN(logger_, "Mobile request list is empty");
      break;
    }

   RequestPtr request_ptr( request_controller_->mobile_request_list_.front());
   request_controller_->mobile_request_list_.pop_front();

    bool init_res = request_ptr->Init();  // to setup specific
                                                    // default timeout

    const uint32_t timeout_in_mseconds = request_ptr->default_timeout();
    RequestInfoPtr request_info_ptr(new MobileRequestInfo(request_ptr,
                                                          timeout_in_mseconds));

    request_controller_->waiting_for_response_.Add(request_info_ptr);
    LOG4CXX_DEBUG(logger_, "timeout_in_mseconds " << timeout_in_mseconds);

    if (0 != timeout_in_mseconds) {
      request_controller_->UpdateTimer();
    } else {
      LOG4CXX_DEBUG(logger_, "Default timeout was set to 0. "
                    "RequestController will not track timeout "
                    "of this request.");
    }

    AutoUnlock unlock(auto_lock);

    // execute
    if ((false == request_controller_->IsLowVoltage()) &&
        request_ptr->CheckPermissions() && init_res) {
      LOG4CXX_DEBUG(logger_, "Execute MobileRequest corr_id = "
                    t_info_ptr->requestId()
                    timeout: " << timeout_in_mseconds);
      request_ptr->Run();
    }
  }
}
```

request_ptr->Run()调用 RPC 内部处理函数

4 SetDisplayLayoutRequest

4.1 通过 SendHMIRequest 转发

```
void SetDisplayLayoutRequest::Run() {
  LOG4CXX_AUTO_TRACE(logger_);
  ApplicationConstSharedPtr app =
      ApplicationManagerImpl::instance()->application(connection_key());

  if (!app) {
    LOG4CXX_ERROR(logger_, "Application is not registered");
    SendResponse(false, mobile_apis::Result::APPLICATION_NOT_REGISTERED);
    return;
  }

  (*message_)[strings::msg_params][strings::app_id] = app->app_id();
  SendHMIRequest(hmi_apis::FunctionID::UI_SetDisplayLayout,
                 &((*message_)[strings::msg_params]), true);

}
```

4.2 SendHMIRequest

```
uint32_t CommandRequestImpl::SendHMIRequest(
    const hmi_apis::FunctionID::eType& function_id,
    const smart_objects::SmartObject* msg_params,
    bool use_events) {
  smart_objects::SmartObjectSPtr result = new smart_objects::SmartObject;

  const uint32_t hmi_correlation_id =
      ApplicationManagerImpl::instance()->GetNextHMICorrelationID();
  if (use_events) {
    LOG4CXX_DEBUG(logger_,
                  "subscribe_on_event " << function_id << " "
                                        << hmi_correlation_id);
    subscribe_on_event(function_id, hmi_correlation_id);
  }

  smart_objects::SmartObject& request = *result;
  request[strings::params][strings::message_type] = MessageType::kRequest;
  request[strings::params][strings::function_id] = function_id;
  request[strings::params][strings::correlation_id] = hmi_correlation_id;
  request[strings::params][strings::protocol_version] =
      CommandImpl::protocol_version_;
  request[strings::params][strings::protocol_type] =
      CommandImpl::hmi_protocol_type_;

  if (msg_params) {
    request[strings::msg_params] = *msg_params;
  }

  if (!ApplicationManagerImpl::instance()->ManageHMICommand(result)) {
    LOG4CXX_ERROR(logger_, "Unable to send request");
    SendResponse(false, mobile_apis::Result::OUT_OF_MEMORY);
  }
  return hmi_correlation_id;
}
```

4.3 ManageHMICommand

```
bool ApplicationManagerImpl::ManageHMICommand(
    const commands::MessageSharedPtr message) {
  LOG4CXX_AUTO_TRACE(logger_);

  if (!message) {
    LOG4CXX_WARN(logger_, "Null-pointer message received.");
    return false;
  }

  if (IsLowVoltage()) {
    LOG4CXX_WARN(logger_, "Low Voltage is active");
    return false;
  }

  MessageHelper::PrintSmartObje

  CommandSharedPtr command = HMICommandFactory::CreateCommand(message);
  if (!command) {
    LOG4CXX_WARN(logger_, "Failed to create command from smart object");
    return false;
  }

  int32_t message_type =
      (*(message.get()))[strings::params][strings::message_type].asInt();

  // Init before adding to request controller to be able to set request timeout
  if (command->Init()) {
    if (kRequest == message_type) {
      LOG4CXX_DEBUG(logger_, "ManageHMICommand");
      request_ctrl_.addHMIRequest(command);
    }

    command->Run();
    if (kResponse == message_type) {
      const uint32_t correlation_id =
          (*(message.get()))[strings::params][strings::correlation_id].asUInt();
      request_ctrl_.OnHMIResponse(correlation_id);
    }
    return true;
  }
  return false;
}
```

HMICommandFactory::CreateCommand工厂模式根据RPC定义的ID创建对应RPC消息处理对象

此处command 为UiSetDisplayLayoutRequest类型

command->Run进行内部处理

4.4 UiSetDisplayLayoutRequest::Run

```
void UiSetDisplayLayoutRequest::Run() {
  LOG4CXX_AUTO_TRACE(logger_);
  SendRequest();
}
```

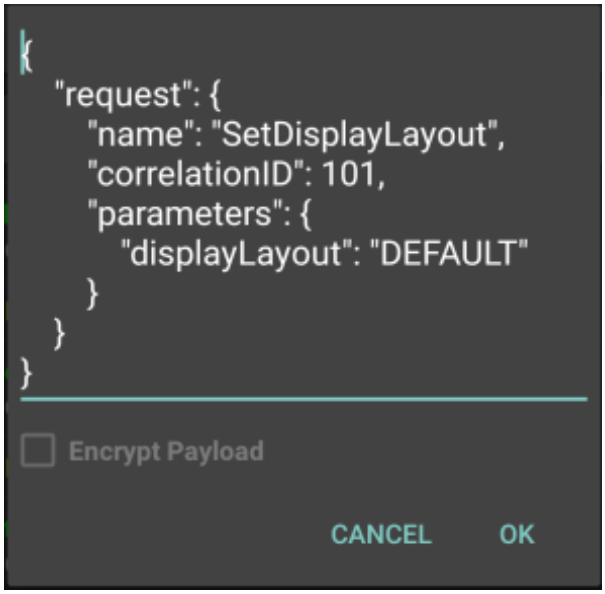
4.5 监视数据

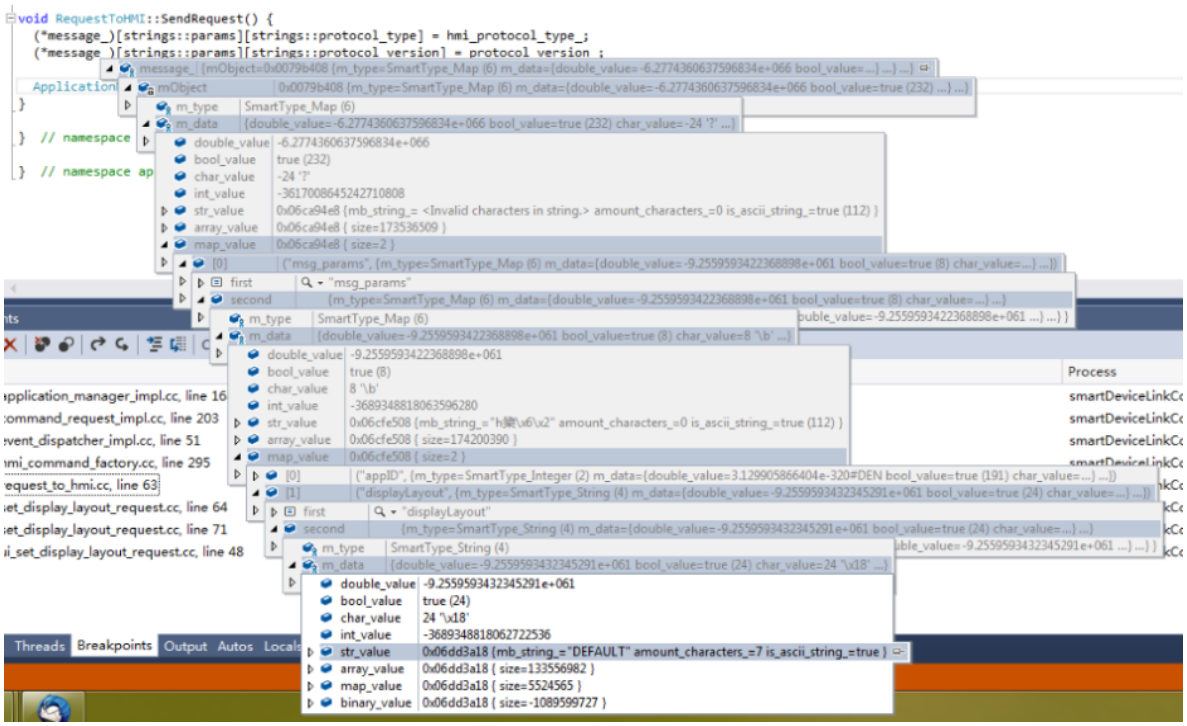
```
void RequestToHMI::SendRequest() {
  (*message_)[strings::params][strings::protocol_type] = hmi_protocol_type_;
  (*message_)[strings::params][strings::protocol_version] = protocol_version_;

  ApplicationManagerImpl::instance()->SendMessageToHMI(message_);
}
```

监视上图 message 的内容，详情见第 5 节。

5 setDisplayLayout 不同参数数据监视

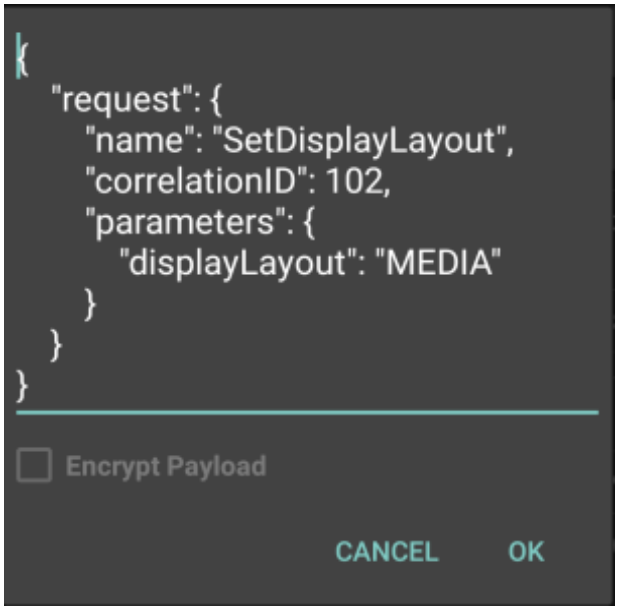
5.1 参数为 DEFAULT

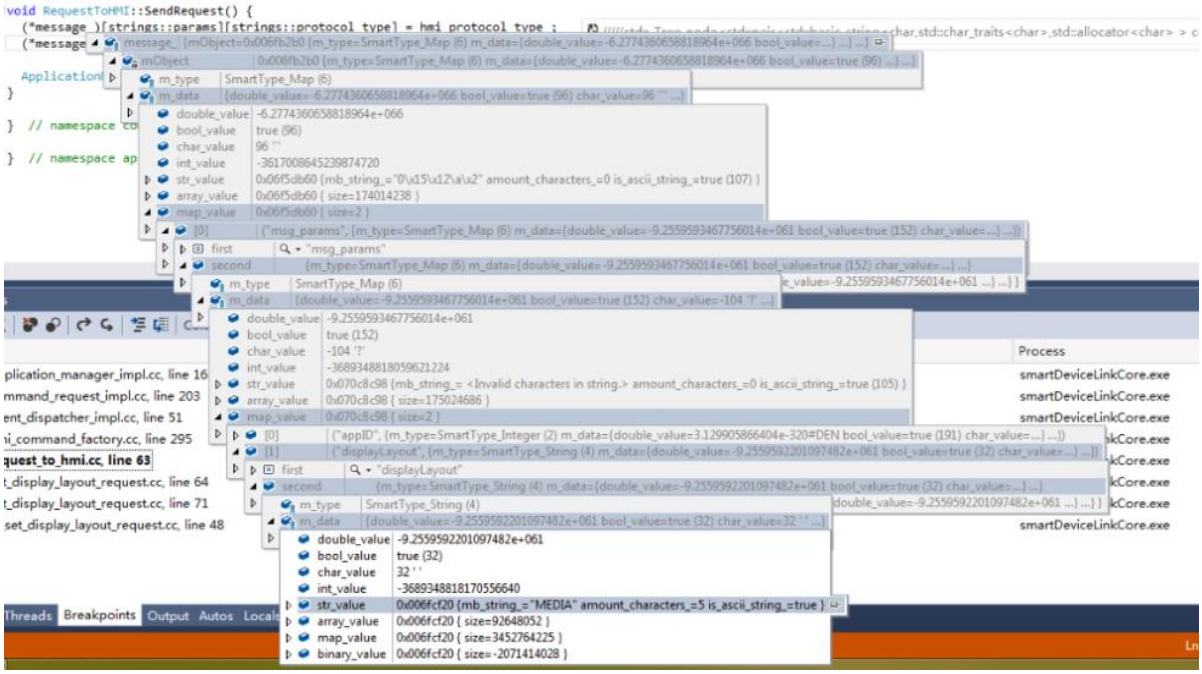




重点关注 mb_string_ = “DEFAULT”。

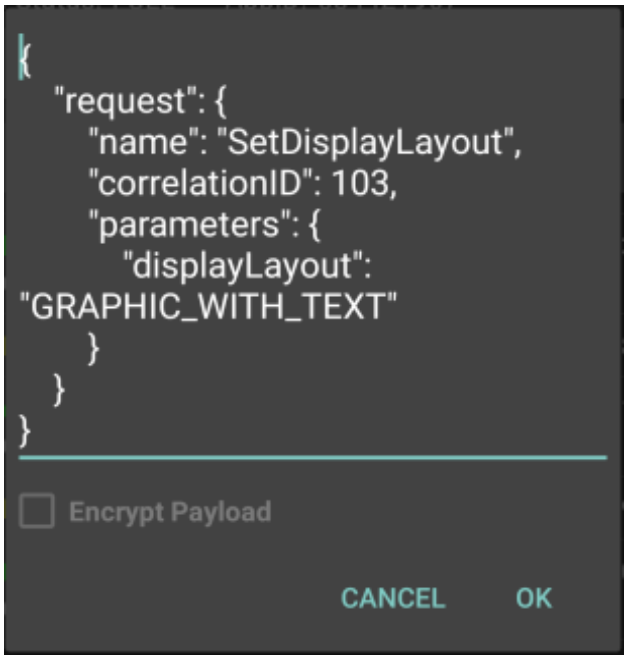
5.2 参数为 MEDIA

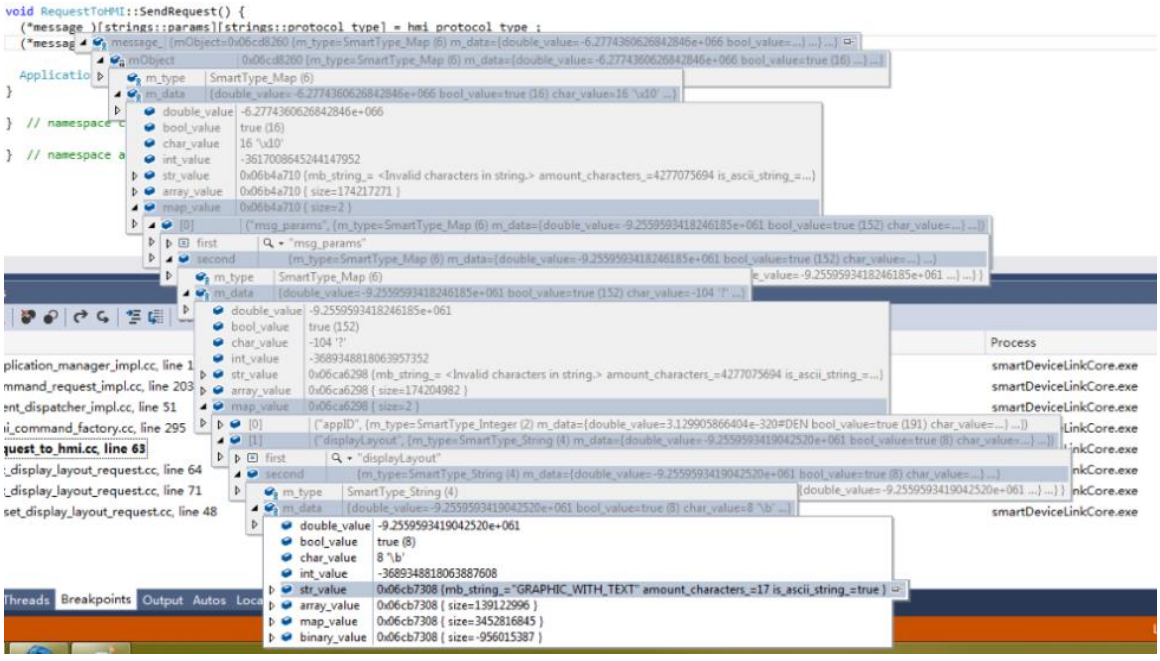




重点关注 mb_string_ = “MEDIA”。

5.3 参数为 GRAPHIC_WITH_TEXT





重点关注 mb_string_ = “GRAPHIC_WITH_TEXT”。

选择其他参数，发送给 HMI 的数据也类似

6 结论

通过流程分析，发现 setDisplayLayout 的 RPC 在 SDL 中并无特殊处理，而且选择不同的参数，发送给 HMI 侧的数据都与之匹配。

因此，setDisplayLayout 在 HMI 表现出来的问题与 SDL 内部无关。