QMP 自定义接口

1. 参考资料

1. QEMU Machine Protocol ==> qmp-intro
2. QEMU Machine Protocol Specification ==> qmp-spec
3. How to write QMP commands using the QAPI framework ==> writing-qmp-commands

2. 定义模式和实现

1. 在 `./qapi/XXX.json` 中定义 `json` 对象， `qemu` 会在 `gcc` 预编译之前生成 `./qapi/qapi-XXX-XXX.h` 文件，注意 `qemu` 这里要求**语法检查特别严格**，包括**说明文档**，**变量名称**等，所以此处要结合参考资料。
2. 在相应的源文件增加函数的实现，这里有几个注意点
    1. 必须以 `qmp_` 开头后面加命令名称，例如 `qmp_helloworld` 为函数名。
    2. 对于没有声明 `return` 的 `json` 对象，对应函数也返回 `void` ，否则一律返回指针。
    3. 在函数内部使用 `g_malloc0(sizeof())` 分配对象，并且不要在函数内 `free` ，因为 `qemu` 会自动生成一个函数去释放对象。
    4. 对于 `*XXX` 这样带有 `*` 的 `json` 对象，表示一个可选择是否传入的对象，这样的对象会对应生成一个 `has_XXX` 的 `bool` 在函数形参，而且每一个 `has_XXX` 和 `XXX` 是对应的，并且顺序也应该按照 `json` 对象的顺序。
    5. 在形参列表的最后一个参数必须传入 `Error **errp` ，这个用于错误处理，使用 `error_setg()` 函数，第一个传入 `Error **errp` 指针，下一个参数是对错误的自然语言描述，这是一个类似 `printf` 的字符串。
3. `qemu` 推荐在定义 `qmp` 接口的同时，将 `hmp` 接口也一同定义
4. 在 `./qapi/aqpi-types-XXX.h` 会定义 `struct json` ，在 `./qapi-command-XXX.h` 会定义 `command json`

3. 出错处理

1. 在有返回值得情况下返回栈内单独创建的对象，若想**返回** `NULL` 则一定要**报错**。否则 `qapi-visit-core.c:47` 会因为 `assert` 而 `Abort`
2. 如果结构体内嵌结构体，则内嵌的结构体也要 `g_malloc0()` 构造出来，否则 `qapi-visit-core.c:47` 会因为 `assert` 而 `Abort`
3. 由上述两条可以得出结论，不需要手动析构一个在栈上创建的局部指针变量。因为返回指针出去后， `qemu` 会自动析构包括内在结构体在内的全部堆区内存。

4. 测试结果

1.
```
# 开启QMP
{ "execute": "qmp_capabilities" }
{"return": {}}

{ "execute": "query_helloThread" }
{"return": {"thread_number": 8}}
{ "execute": "set_helloThread", 'arguments': {'value': 1 } }
{"return": {"thread_number": 1}}
# 增加出错处理
{ "execute": "set_helloThread", 'arguments': {'value': -1 } }
```

```
{"error": {"class": "GenericError", "desc": "value must be a positive
number"}}
{ "execute": "set_helloThread" }
{"return": {"thread_number": 0}}

# 未使用出错处理
{ "execute": "set_helloThread" }
# qemu-system-x86_64: qapi/qapi-visit-core.c:47: visit_start_struct:
Assertion `!(v->type & VISITOR_OUTPUT) || *obj' failed.
#Aborted
```