# DataGridView动态生成

DataGridViewEdit^ mDataGridView;

Void GF\_Main::initDataGridView(void) {

mDataGridView = gcnew DataGridViewEdit(toolStripButton\_Hex->Checked);

mDataGridView->Dock = DockStyle::Fill;

mDataGridView->EditingControlShowing += gcnew DataGridViewEditingControlShowingEventHandler(this, &GF\_Main::dataGridView1\_EditingControlShowing);

mDataGridView->OnRegisterValueChanged += gcnew DataGridViewEdit::registerValueChangeHandler(this, &GF\_Main::dataGridView\_RegisterAdjust);

mDataGridView->CellEndEdit += gcnew DataGridViewCellEventHandler(this, &GF\_Main::dataGridView1\_CellEndEdit);

mDataGridView->CellClick += gcnew DataGridViewCellEventHandler(this, &GF\_Main::dataGridView1\_CellClick);

mDataGridView->CellValueChanged += gcnew DataGridViewCellEventHandler(this, &GF\_Main::dataGridView1\_CellValueChanged);

mDataGridView->CellMouseEnter += gcnew DataGridViewCellEventHandler(this, &GF\_Main::dataGridView1\_CellMouseEnter);

panel\_DataGridView->Controls->Add(mDataGridView);

mDataTable = gcnew DataTable;

// automatically generate the DataGridView columns.

mDataGridView->AutoGenerateColumns = true;

// full row select

// mDataGridView->SelectionMode = DataGridViewSelectionMode::FullRowSelect;

// set up the data source.

bindingSource1->DataSource = mDataTable;

mDataGridView->DataSource = bindingSource1;

bindingNavigator1->BindingSource = bindingSource1;

// automatically resize the visible rows.

mDataGridView->AutoSizeRowsMode = DataGridViewAutoSizeRowsMode::DisplayedCells;

mDataGridView->AutoSizeColumnsMode = DataGridViewAutoSizeColumnsMode::Fill;

// disable edit by click

// mDataGridView->EditMode = DataGridViewEditMode::EditProgrammatically;

// only vertical scroll bar

mDataGridView->ScrollBars = ScrollBars::Vertical;

// hide row header

mDataGridView->RowHeadersVisible = false;

// forbid add new rows

mDataGridView->AllowUserToAddRows = false;

array<Object^, 2>^ columns = gcnew array<Object^, 2>{

// Name Type Visible

{ COL\_NAME, String::typeid, true },

{ COL\_VALUE, String::typeid, true },

{ COL\_WIDTH, String::typeid, true },

{ COL\_R\_W, String::typeid, true },

{ COL\_REMARK, String::typeid, true },

{ COL\_SELECT, String::typeid, true },

{ COL\_PAGENAME, String::typeid, true },

{ COL\_INDEX, Int32::typeid, false },

};

Table::makeDataTable(mDataGridView, mDataTable, columns);

for each (DataGridViewColumn^ column in mDataGridView->Columns) {

if (column->Name == COL\_VALUE) {

mDataGridView->Columns[COL\_VALUE]->CellTemplate->Style->BackColor = Color::Wheat;

} else {

column->ReadOnly = true;

}

}

}

# 获取选择单元格的行

//下面是获取当前行

DataGridViewRow^ viewRow = mDataGridView->CurrentCell->OwningRow;

或者

DataGridViewRow^ viewRow = dataGridView->Rows[e->RowIndex];

//下面是读取指定行中某列

String^ pageName = viewRow->Cells[COL\_PAGENAME]->Value->ToString();

// 下面是判断是不是数值（应该是十六进制）

String^ sValue = viewRow->Cells[COL\_VALUE]->Value->ToString();

UInt32 value;

if (mDataGridView->hexStyle) {

value = UInt32::Parse(sValue, NumberStyles::HexNumber);

} else {

value = UInt32::Parse(sValue);

}