Subject: 2023.1 alpha
Posted by mirek on Mon, 01 May 2023 17:44:56 GMT
View Forum Message <> Reply to Message

All things planned are ready for 2023.1:

- MacOS is supported again; POSIX and MACOS releases are now merged to single archive
- clang-format integration
- .iml image selection database tool
- many fixes....

I am therefore designating current nightly as "2023.1 alpha"....

Subject: Re: 2023.1 alpha
Posted by GiuMar on Mon, 01 May 2023 19:40:15 GMT
View Forum Message <> Reply to Message

Great.
Many thanks.

Subject: Re: 2023.1 alpha
Posted by zsolt on Tue, 02 May 2023 05:35:15 GMT
View Forum Message <> Reply to Message

Thanks!

Subject: Re: 2023.1 alpha
Posted by pvictor on Tue, 02 May 2023 06:11:59 GMT
View Forum Message <> Reply to Message

Hi

Please correct this issue https:// www.ultimatepp.org/forums/index.php?t=msg&th=12005&s tart=0& in the new release.

Best regards,
Victor

Subject: Re: 2023.1 alpha
Posted by BioBytes on Tue, 02 May 2023 06:41:20 GMT
View Forum Message <> Reply to Message

Hello,
Many thanks Mirek for this new release

Subject: Re: 2023.1 alpha
Posted by mirek on Tue, 02 May 2023 18:14:21 GMT
View Forum Message <> Reply to Message

pvictor wrote on Tue, 02 May 2023 08:11Hi

Please correct this issue https:// www.ultimatepp.org/forums/index.php?t=msg&th=12005&s tart=0& in the new release.

Best regards,
Victor

Thanks, hopefully fixed (although I am still puzzled with that format variation). Tried to fill some more holes too.

Mirek

Subject: Re: 2023.1 alpha
Posted by Tom1 on Fri, 05 May 2023 12:51:20 GMT
View Forum Message <> Reply to Message

Hi Mirek,

Thanks! Pleased to hear release is approaching. I will keep testing.

For now, here's a couple of compilation warnings on MSBT22x64:

C:\upp-git\upp.src\uppsrc\plugin\sqlite3\Sqlite3upp.cpp(644): warning C4244: 'argument': conversion from '__int64' to 'int', possible loss of data
C:\upp-git\upp.src\uppsrc\Painter\Painter.cpp(270): warning C4267: 'argument': conversion from 'size_t' to 'int', possible loss of data
Best regards,

Tom

Subject: Re: 2023.1 alpha
Posted by Tom1 on Fri, 12 May 2023 08:23:39 GMT
View Forum Message <> Reply to Message

Hi Mirek,

There is an issue with FileSel with the current version. I can select a file, e.g. "datafile.csv", but the returned result is "datafile.csv.csv". Sometimes FileSel compalains it cannot find the file that was picked up from the list:

```
 FileSel fs;
 fs.Multi();
 fs.Type(t_("All supported formats"),"*.csv; *.CSV; *.txt; *.TXT");
 fs.Type(t_("Comma separated values file"),"*.csv; *.CSV");
 fs.Type(t_("Text file"),"*.txt; *.TXT");

 if(fs.ExecuteOpen(t_("Import data file"))){
```

Best regards,

Tom

Subject: Re: 2023.1 alpha
Posted by Tom1 on Fri, 12 May 2023 13:09:59 GMT
View Forum Message <> Reply to Message

Hi,

Would it be possible to add Null support for float?

I think below you can find pretty much what is needed in Core/Defs.h:

```
...
constexpr double DOUBLE_NULL    =    -std::numeric_limits<double>::infinity();
constexpr float FLOAT_NULL    =    -std::numeric_limits<float>::infinity();

class Nuller {
public:
 operator int() const              { return INT_NULL; }
 operator int64() const             { return INT64_NULL; }
 operator double() const              { return DOUBLE_NULL; }
 operator float() const             { return FLOAT_NULL; }
 operator bool() const              { return false; }

 Nuller() {}
};

extern const Nuller Null;

template <class T> void SetNull(T& x) { x = Null; }

template <class T> bool IsNull(const T& x)       { return x.IsNullInstance(); }
```

```
template<> inline bool  IsNull(const int& i)     { return i == INT_NULL; }
template<> inline bool  IsNull(const int64& i)   { return i == INT64_NULL; }
template<> inline bool  IsNull(const double& r)  { return !(std::abs(r) <
std::numeric_limits<double>::infinity()); }
template<> inline bool  IsNull(const float& r)   { return !(std::abs(r) <
std::numeric_limits<float>::infinity()); }
template<> inline bool  IsNull(const bool& r  )  { return false; }
...
```

Best regards,

Tom

Subject: Re: 2023.1 alpha
Posted by mirek on Sat, 13 May 2023 15:47:16 GMT
View Forum Message <> Reply to Message

Tom1 wrote on Fri, 12 May 2023 10:23Hi Mirek,

There is an issue with FileSel with the current version. I can select a file, e.g. "datafile.csv", but the returned result is "datafile.csv.csv". Sometimes FileSel compalains it cannot find the file that was picked up from the list:

```
 FileSel fs;
 fs.Multi();
 fs.Type(t_("All supported formats"),"*.csv; *.CSV; *.txt; *.TXT");
 fs.Type(t_("Comma separated values file"),"*.csv; *.CSV");
 fs.Type(t_("Text file"),"*.txt; *.TXT");

 if(fs.ExecuteOpen(t_("Import data file"))){
```

Best regards,

Tom

Remove ';'.

```
  fs.Type(t_("All supported formats"),"*.csv *.CSV *.txt *.TXT");
  fs.Type(t_("Comma separated values file"),"*.csv *.CSV");
  fs.Type(t_("Text file"),"*.txt; *.TXT");
```

Subject: Re: 2023.1 alpha

Posted by mirek on Sat, 13 May 2023 15:48:07 GMT
View Forum Message <> Reply to Message

Tom1 wrote on Fri, 12 May 2023 15:09Hi,

Would it be possible to add Null support for float?

I think below you can find pretty much what is needed in Core/Defs.h:

```
...
constexpr double DOUBLE_NULL    =    -std::numeric_limits<double>::infinity();
constexpr float FLOAT_NULL    =    -std::numeric_limits<float>::infinity();

class Nuller {
public:
 operator int() const              { return INT_NULL; }
 operator int64() const             { return INT64_NULL; }
 operator double() const              { return DOUBLE_NULL; }
 operator float() const             { return FLOAT_NULL; }
 operator bool() const              { return false; }

 Nuller() {}
};

extern const Nuller Null;

template <class T> void SetNull(T& x) { x = Null; }

template <class T> bool IsNull(const T& x)       { return x.IsNullInstance(); }

template<> inline bool  IsNull(const int& i)     { return i == INT_NULL; }
template<> inline bool  IsNull(const int64& i)   { return i == INT64_NULL; }
template<> inline bool  IsNull(const double& r)  { return !(std::abs(r) <
std::numeric_limits<double>::infinity()); }
template<> inline bool  IsNull(const float& r)   { return !(std::abs(r) <
std::numeric_limits<float>::infinity()); }
template<> inline bool  IsNull(const bool& r  )  { return false; }
...
```

Best regards,

Tom

Too big change in time of release, even if I thought this is a good idea (I am yet undecided). Let us discuss float at the start of the next cycle, ok?

Subject: Re: 2023.1 alpha
Posted by Tom1 on Mon, 15 May 2023 06:38:19 GMT

mirek wrote on Sat, 13 May 2023 18:47Tom1 wrote on Fri, 12 May 2023 10:23Hi Mirek,

There is an issue with FileSel with the current version. I can select a file, e.g. "datafile.csv", but the returned result is "datafile.csv.csv". Sometimes FileSel compalains it cannot find the file that was picked up from the list:

```
FileSel fs;
fs.Multi();
fs.Type(t_("All supported formats"),"*.csv; *.CSV; *.txt; *.TXT");
fs.Type(t_("Comma separated values file"),"*.csv; *.CSV");
fs.Type(t_("Text file"),"*.txt; *.TXT");

if(fs.ExecuteOpen(t_("Import data file"))){
```

Best regards,

Tom

Remove ';'.

```
fs.Type(t_("All supported formats"),"*.csv *.CSV *.txt *.TXT");
fs.Type(t_("Comma separated values file"),"*.csv *.CSV");
fs.Type(t_("Text file"),"*.txt; *.TXT");
```

Thanks Mirek!

For some unknown reason I never knew there was no need for this semicolon.

Best regards,

Tom

Subject: Re: 2023.1 alpha
Posted by Tom1 on Mon, 15 May 2023 06:43:34 GMT

mirek wrote on Sat, 13 May 2023 18:48Tom1 wrote on Fri, 12 May 2023 15:09Hi,

Would it be possible to add Null support for float?

I think below you can find pretty much what is needed in Core/Defs.h:

```
...
constexpr double DOUBLE_NULL    =    -std::numeric_limits<double>::infinity();
constexpr float FLOAT_NULL    =    -std::numeric_limits<float>::infinity();
```

```
class Nuller {
public:
 operator int() const              { return INT_NULL; }
 operator int64() const             { return INT64_NULL; }
 operator double() const              { return DOUBLE_NULL; }
 operator float() const             { return FLOAT_NULL; }
 operator bool() const              { return false; }

 Nuller() {}
};

extern const Nuller Null;

template <class T> void SetNull(T& x) { x = Null; }

template <class T> bool IsNull(const T& x)       { return x.IsNullInstance(); }

template<> inline bool  IsNull(const int& i)     { return i == INT_NULL; }
template<> inline bool  IsNull(const int64& i)   { return i == INT64_NULL; }
template<> inline bool  IsNull(const double& r)  { return !(std::abs(r) <
std::numeric_limits<double>::infinity()); }
template<> inline bool  IsNull(const float& r)   { return !(std::abs(r) <
std::numeric_limits<float>::infinity()); }
template<> inline bool  IsNull(const bool& r  )  { return false; }
...
```

Best regards,

Tom

Too big change in time of release, even if I thought this is a good idea (I am yet undecided). Let us discuss float at the start of the next cycle, ok?

Yes, absolutely. After this release is just fine. The need for float Null just pops up every once in a while and does not seem so ground breaking from my point of view.

Thanks and best regards,

Tom