create plugin instance

```
NPError
NPP_New(

      NPMIMEType pluginType,
      NPP instance,
      uint16 mode,
      int16 argc,
      char* argn[],
      char* argv[],
      NPSavedData* saved)

{

      int i;
      PluginInstance* This;

      /* create instance data */
      instance->pdata = NPN_MemAlloc(sizeof(PluginInstance));
      memset(instance->pdata, 0, sizeof(PluginInstance));
      This = (PluginInstance*) instance->pdata;

      /* examine the parameter list */
      for (i=0; i<argc; i++) {
          if (strcmp(argn[i], "target") == 0) copy_param(&This->target,
       argv[i]);
          if (strcmp(argn[i], "input") == 0) copy_param(&This->input,
       argv[i]);
    }
      /* Create a temporary filename, where we can save the rexx-source */
      tmpnam(This->tmpFileName);
      This->tmpFile = creat(This->tmpFileName, S_IWRITE | S_IREAD);

      return NPERR_NO_ERROR;

}
```

destroy plugin instance

```
NPError
NPP_Destroy(

    NPP instance,
    NPSavedData** save)

{

    /* get a reference to our private instance data */
    PluginInstance* This = (PluginInstance*) instance->pdata;

    if (This != NULL) {
        /* delete the temporary file, which was created in NPP_New
     */
        unlink(This->tmpFileName);

        /* Free the allocated memory of our private instance data */
        NPN_MemFree(This->target);
        NPN_MemFree(This->input);
        NPN_MemFree(instance->pdata);
        instance->pdata = NULL;
   }
    return NPERR_NO_ERROR;

}
```

initialize stream

```
NPError
NPP_NewStream(

        NPP instance,
        NPMIMEType type,
        NPStream *stream,
        NPBool seekable,
        uint16 *stype)

{

        return NPERR_NO_ERROR;

}
```

set block size

```
int32
NPP_WriteReady(

        NPP instance,
        NPStream *stream)

{

        return STREAMBUFSIZE;

}
```

receive data

```
int32
NPP_Write(

        NPP instance,
        NPStream *stream,
        int32 offset,
        int32 len,
        void *buffer)

{

        PluginInstance* This = (PluginInstance*) instance->pdata;
        return write(This->tmpFile, (char*)buffer, (size_t)(len<=STREAMBUFSIZE
     ? len : STREAMBUFSIZE));

}
```

destroy stream

```
NPError
NPP_DestroyStream(

        NPP instance,
        NPStream *stream,
        NPError reason)

{

        PluginInstance* This = (PluginInstance*) instance->pdata;
        if (stream != This->rexx_stream) {
            close(This->tmpFile);
            return rexx_exec(instance);
     }

}
```

REXX handle SAY-Command

```
APIRET handle_RXSIOSAY( PEXIT ParmBlock )
{

        NPP instance;
        PluginInstance* This;
        void** UserDataAddr;
        USHORT Flag;

        UserDataAddr = NPN_MemAlloc(8);
        RexxQueryExit("hook_RXSIOSAY", NULL, &Flag, (PUCHAR)UserDataAddr);
        instance = (NPP)*UserDataAddr;
        This = (PluginInstance*)instance->pdata;
        if (This->rexx_stream == NULL)
            NPN_NewStream(instance, "text/html", This->target, &This-
          >rexx_stream);

     NPN_Write(
            instance,
            This->rexx_stream,
            ((EXIT*)ParmBlock)->siosay.rxsio_string.strlength,
            ((EXIT*)ParmBlock)->siosay.rxsio_string.strptr);

     NPN_MemFree(UserDataAddr);
        return RXEXIT_HANDLED;

}
```

REXX system exit handler

```
APIRET APIENTRY hook_RXSIOSAY(

        LONG ExitNumber,
        LONG Subfunction,
        PEXIT ParmBlock)

{

        switch (ExitNumber) {
        case RXSIO:
            switch (Subfunction) {
                case RXSIOSAY: return handle_RXSIOSAY(ParmBlock);
                default: return RXEXIT_NOT_HANDLED;
          }
     default: return RXEXIT_NOT_HANDLED;
        }

}
```

REXX execute rexx interpreter

```
NPError
rexx_exec(NPP instance)
{

        PluginInstance* This;
        long UserDataAddr;
        SHORT ReturnCode;
        RXSTRING Result;
        RXSYSEXIT sysexit[2];
        RXSTRING arglist[1];

        This = (PluginInstance*) instance->pdata;
        UserDataAddr = (long)instance;

        sysexit[0].sysexit_code = RXSIO;
        sysexit[0].sysexit_name = "hook_RXSIOSAY";
        sysexit[1].sysexit_code = RXENDLST;

        arglist[0].strptr = This->input;
        arglist[0].strlength = strlen(This->input);

        RexxRegisterExitExe("hook_RXSIOSAY", hook_RXSIOSAY, (PUCHAR)&UserDataAddr)
        RexxStart(1, arglist, This->tmpFileName, NULL, NULL, RXCOMMAND,
     sysexit, &ReturnCode, &Result);
        RexxDeregisterExit("hook_RXSIOSAY", NULL);

        if (This->rexx_stream != NULL)
            NPN_DestroyStream(instance, This->rexx_stream, NPRES_DONE);

     return NPERR_NO_ERROR;

}
```