VB Decompiler

Kullanım kılavuzu

Türkçe

Visual Basic 5.0/6.0 · .NET

DotFix Software

İçindekiler

1 Açıklama 4

2 Hızlı başlangıç 5

3 Ana pencere 6

4 Kodu görüntüleme 8

5 Yapay zekâ ile kod iyileştirme 11

6 Açma ve kaydetme 15

6.1 Dosya açma 15

6.2 Analiz veritabanlarıyla çalışma 16

6.3 Yordam listesini kaydetme 17

7 Program işlevleri 18

7.1 Adrese gitme 18

7.2 Dize arama 19

7.3 Dize başvuruları 20

7.4 Belirtilen adresten geri derleme 22

7.5 Verileri değiştirme 23

7.6 Kod karartma 25

8 Program seçenekleri 26

8.1 Program seçenekleri 26

8.2 Formlar 27

8.3 Geri derleme 28

8.4 Disassembler 30

8.5 Renk düzeni 31

8.6 Gelişmiş özellikler . . . 33

8.7 Dil seçimi . . . 37

9 Eklentiler . . . 38

10 Formların görsel olarak yeniden oluşturulması . . . 42

11 İzleme . . . 46

12 Analiz özellikleri . . . 48

13 Kod karartma sınırlamaları . . . 50

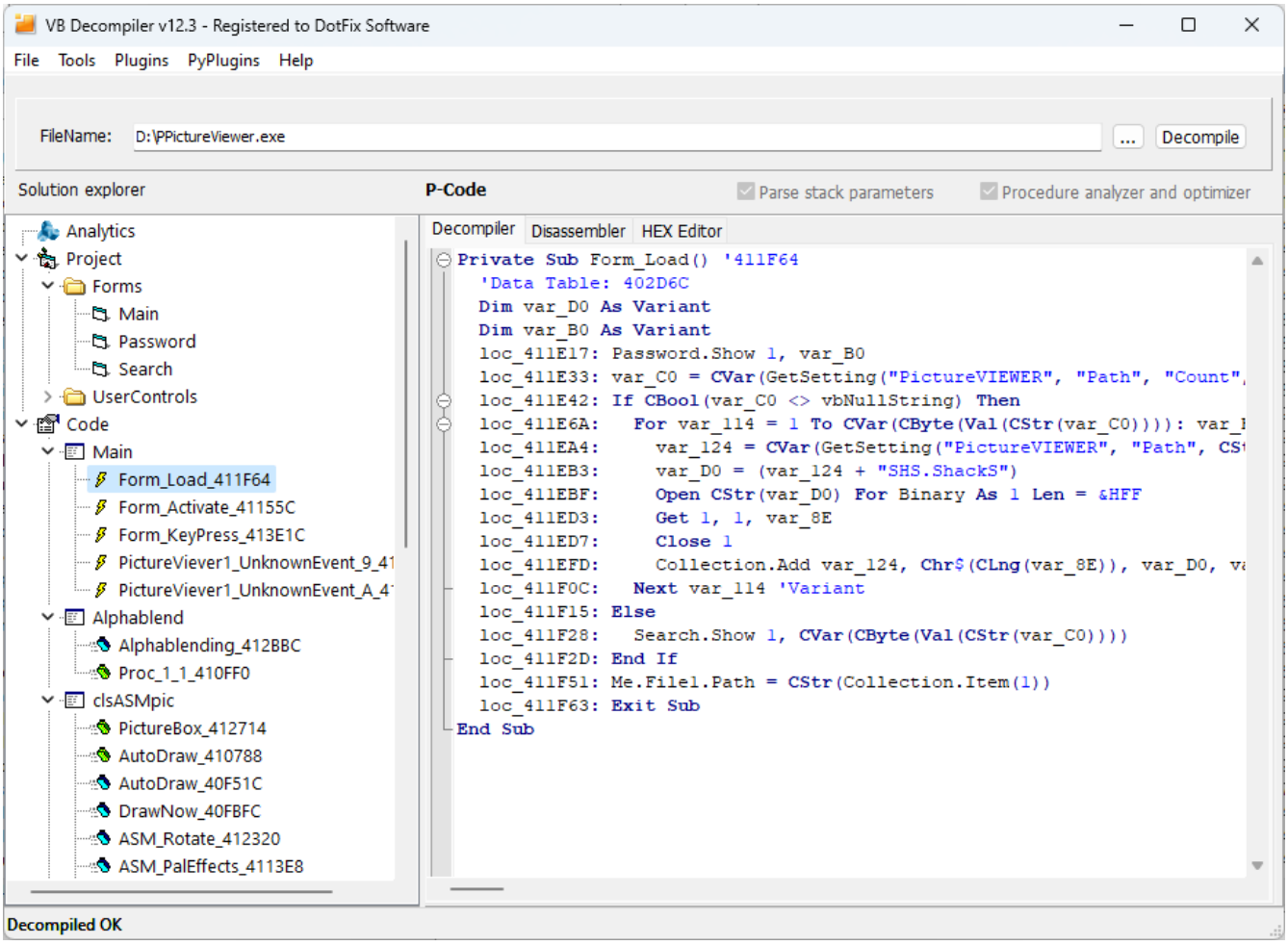
14 Teknik destek . . . 52

15 Kayıt . . . 52

16 Geliştirici hakkında . . . 53

1 Açıklama

VB Decompiler, Visual Basic 5.0/6.0 programları ve .NET assemblyleri için ters mühendislik ve kaynak kodu kurtarma aracıdır. Yüksek seviyeli geri derleme, düşük seviyeli makine kodu çözümlemesi, meta veri incelemesi, tümleşik HEX Düzenleyici ve görsel arayüz yeniden inşasını birleştirir. Tipik kullanım alanları arasında eski yazılım bakımı, olay yanıtı, kötü amaçlı yazılım analizi ve belgelendirilmemiş uygulama davranışının araştırılması yer alır.



Desteklenen kod türleri

- **VB5/6 P-Code:** derleyicinin P-Code komutları ve çalışma zamanı işlemlerinden Visual Basic ifadelerini yeniden inşa eder.
- **VB5/6 Native Code:** x86 makine komutlarını çözümler ve içsel bir emülatör, çalışma zamanı imzaları ve sezgisel yöntemler kullanarak Visual Basic benzeri kodu yeniden inşa eder.
- **.NET:** Ortak Ara Dil (CIL, MS IL olarak da adlandırılır) komutlarını çözümler ve desteklenen IL kalıplarını C# benzeri koda geri derler; orijinal .NET kaynak dilinden bağımsız olarak. .NET derlemelerindeki yerel x86 ve x64 kodunun da makine kodu çözümlemesi yapılabilir.

.NET analiz motoru, kod analizi veya görsel form yeniden inşası için .NET çalışma zamanına veya Reflection'a ihtiyaç duymaz.

Analiz araçları

Davranışı araştırmak için yordam ve metin arama, çapraz referanslar ve değişken yeniden adlandırma ve String Referansları kullanın. Görsel görüntüleyici, derlenmiş BAML kaynaklarında saklanan klasik VB formlarını, .NET Windows Forms'ı ve WPF pencerelerini, sayfalarını ve kullanıcı kontrollerini yeniden inşa eder.

Tracing özellikleri içeren lisanslar, VB5/6 Yerel Kodu ve P-Code'un adım adım emülasyonunu sağlar. Analytic özellikleri içeren lisanslar, VB5/6 ve .NET uygulamaları için kategorize edilmiş raporlar sağlar. YZ yardım (AI Helper), Ollama veya OpenAI API entegrasyonu kullanarak kurtarılan VB5/6 veya C# kodunu işleyebilir.

Kurtarılan kodu yorumlama

Derleme, yorum satırları, birçok yerel değişken adı ve bazı tür ve yordam meta verileri gibi bilgileri kaldırır. Optimizasyon ve kod karartma kod yapısını daha da değiştirebilir. Yeniden oluşturulan kod listesi analiz ve kaynak kodu kurtarma amacıyla kullanılır. Önemli sonuçları makine kodu çözümlemesi, meta veri, string referansları ve ilgili yordamların koduyla doğrulayın.

2 Hızlı başlangıç

1. **Dosya > Programı Aç (File > Open program)**'yi seçin ve analiz edilecek EXE, DLL veya OCX dosyasını açın. İşleme otomatik olarak başlar. Proje ağacının doldurulmasını ve işlemenin tamamlanmasını bekleyin.
2. **Çözüm Gezgini (Solution explorer)**'de bir yordamı açın. **Decompiler** ve **Disasm. (Disassembler)** görünümlerini karşılaştırın; ilgili baytları incelemek için **HEX Editörü (HEX Editor)**'yi kullanın.
3. **Ctrl+T**'e basarak yordamın adına veya adının bir kısmına göre arama yapın. Gösterilen metni aramak için **Ctrl+F**'yi veya dizeleri ve bunları kullanan yordamlara bakmak için **Araçlar > Dize referansları (Tools > String references)**'yi kullanın.
4. Kurtarılmış bir formu açın ve görsel önizleme için **Tasarımcı (salt okunur) (Designer (read only))**'yi seçin. Klasik VB, Windows Forms ve WPF için kurtarılmış olayları kullanarak ilgili işleyici koduna gidin.
5. Daha sonra devam etmek veya sonuçları başka bir analistle paylaşmanız gerektiğinde **Dosya > Veritabanı (File > Data Base)** üzerinden analizi kaydedin.

Bir geri derleme modu seçme

Hızlı Geri derleme, yordamları açtığınızda işler. Private olarak bildirilmiş yordamların imzaları için ek ön analiz yapmak üzere **Prototipleri analiz et (Analyze prototypes)**'yi etkinleştirin. Tüm geri derlenmiş metni aramak veya tam bir proje dışa aktarmak için hızlı geri derlemeyi devre dışı bırakın ve dosyayı tekrar işleyin.

Kurtarılmış kaynağın kaydedilmesi

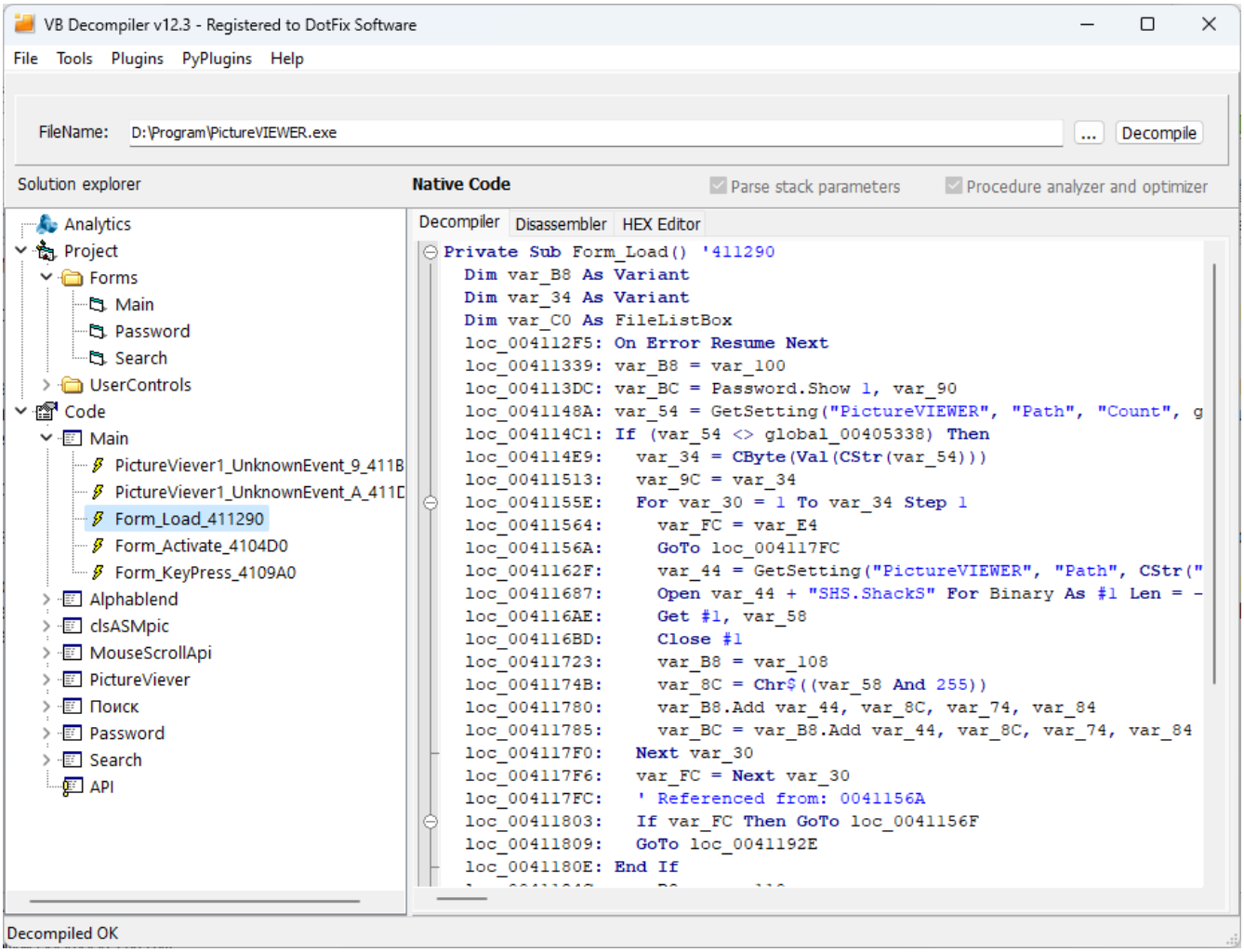
VB5/6 geri derleme sonuçlarını dışa aktarmak için **Dosya > Geri derlenmiş projeyi kaydet (File > Save decompiled project)**'yi seçin ve bir çıktı klasörü belirleyin. Dışa aktarma, kurtarılmış formları ve modülleri içerir; FRX dosyaları ilişkili görselleri ve diğer ikili form verilerini tutar.

Dışa aktarılan kodu bir IDE'de açmadan önce, kurtarılan kaynak koddaki tüm geri derleme hatalarını ve uyumsuzlukları elle düzeltin.

Ayrıntılı iş akışları için Görsel form yeniden inşası, Kod görüntüleme ve Yapay zekâ ile kod iyileştirme'yı görün.

3 Ana pencere

Çözüm Gezgini (Solution explorer), geri kazanılan projeyi formlara, modüllere, sınıflara ve yordamlar halinde gösterir. Yanındaki görünüm seçili kodu, formu veya meta veriyi gösterir. **Visual Studio'daki gibi pencere stili (Window style as in Visual Studio)** seçeneği ağacı sağa ve kodu sola yerleştirir.



Kod ve veri görünümleri

- **Decompiler:** yüksek seviyede yeniden yapılandırılmış kod.
- **Disasm. (Disassembler):** karşılık gelen x86/x64 komutları, VB P-Kodu veya .NET IL ile adresler.
- **HEX Editörü (HEX Editor):** altındaki dosya baytları, sanal adres veya dosya ofseti ile gezinme.
- **Tasarımcı (salt okunur) (Designer (read only)):** desteklenen durumlarda yeniden yapılandırılmış bir form ve geri kazanılan özellikleri.
- **YZ yardım (AI Helper):** şu anda geri derlenen yordam için isteğe bağlı yapay zeka işleme.

Mevcut sekmeler seçili nesneye bağlıdır. Kod türü göstergesi **P-Code**, **Native Code** ve **.NET Assembly**'i ayırt eder.

Yığın parametrelerini geri derle (Parse stack parameters)

P-Kodu için bu kontrol, yığın işlemlerini ve çağrıları argümanlı ifadelerle birleştirir. Kapatılması ara işlemleri ortaya çıkarır ve bilinmeyen bir prototip beklenmedik yığın değerleri tükettiğinde yardımcı olabilir. Native Code için etkinleştirilmesi emülatörü kullanarak daha yüksek seviyeli ifadeleri yeniden yapılandırmak için kullanılır; kapatılması ise assembly seviyesinde çıktıyı ortaya çıkarır. .NET için, yığın işleme IL'yi C# benzeri ifadeler olarak yeniden yapılandırmaya katkıda bulunur.

Prosedür Analizörü ve Optimize Edici (Procedure analyzer and optimizer)

Bu kontrol, ifadeleri ve tanınan kontrol akışı desenlerini birleştirerek tanınan derleyici yardımcı kodunu kaldırarak yeniden yapılandırılmış listeyi basitleştirir. VB5/6 ve .NET koduna uygulanır. .NET optimizatörü, `while`, `do...while` ve `foreach` desenlerini, iç içe koşulları, operatör ifadelerini ve interpolasyonlu-dize işleyici desenlerini yeniden yapılandırmak için kullanılabilir. Bu dönüşümler geri derleyici tarafından gerçekleştirilir ve YZ yardım gerektirmez.

Bir ifade veya dal belirsizse, çıktıyı optimizasyon kapalı ile karşılaştırın ve makine kodu çözümlemesini inceleyin. **Hızlı Geri Derleme (Fast decompilation)** modunda önbellekleme olmadan, bu kontroller yordamı incelerken değiştirilebilir. Tam geri derleme veya önbelleklenen çıktı ile, ayarları değiştirdikten sonra dosyayı tekrar işleyin.

Gezinme ve klavye erişimi

Ağacın üzerindeki ok düğmeleri daha önce görüntülenen yordamlar arasında gezinir; **Esc** önceki yordama döner. Mouse Geri ve İleri düğmeleri, yordam içinde hatırlanan konumu da içeren aynı geçmişi kullanır.

Alt+1'e basarak Çözüm Gezgini ile aktif görünüm arasında odak değiştirin. Ağaçta **Enter**'ye basarak seçili nesneyi açın. HEX Düzenleyici'de, **Tab** onaltılık ve metin alanları arasında hareket eder ve ardından sonraki kontrole gider. Arayüz etiketleri ekran-okuyucu gezinmesi için giriş alanlarıyla ilişkilendirilmiştir.

Bağlama duyarlı kısayollar, çapraz referanslar ve yeniden adlandırma için Kodun görüntülenmesi'na bakın ve Analitik görünüm için Analitik özellikler'e bakın.

4 Kodu görüntüleme

Kod görünümleri, sözdizimi vurgulaması, adres tabanlı gezinme ve çapraz referansları birleştirir. Bir yordamı anlamak için yüksek seviyeli kod listesini kullanın; sonucu doğrulamak için makine kodu çözümlemesini ve dosya baytlarını inceleyin.

Tanımlayıcıları vurgulama ve bunlara gitme

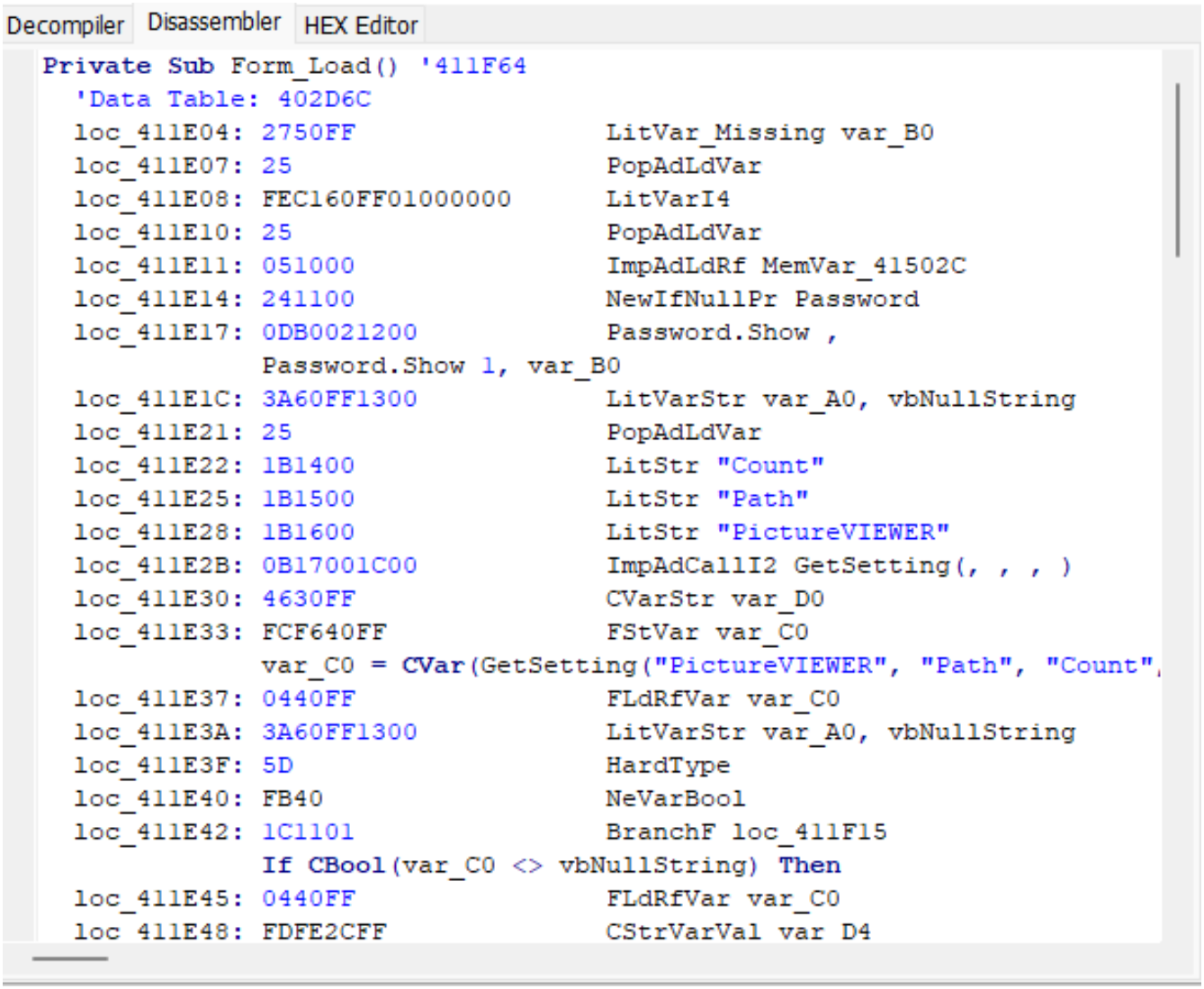
İmleci bir yazmaç, değişken veya fonksiyon adı üzerine yerleştirerek mevcut listedeki eşleşen oluşumları vurgulayın. `var_XX` ve `str_XX` gibi yerel değişkenler, üye ifadesinin tabanı olarak kullanıldığında da vurgulanır; örneğin `str_1.Value`. Bir `global_XX` belirleyicisi seçildiğinde, bu belirleyici mevcut yordam içindeki oluşumları vurgular; bu vurgulama proje çapındaki bir referans aramasından ayrıdır.

Tanınan bir yordam çağrısına çift tıklayarak çağrılan yordamı açın. Adres benzeri sabitler yerine HEX Düzenleyicisi'ne karşılık gelen baytlara yönlendirilebilir. **Esc**, gezinme oklarını veya fare Back/İleri düğmelerini kullanarak görüntüleme geçmişinde gezinin.

Geri derlenmiş ve makine kodu çözümlenmiş kodu birleştirme

Geri derlenmiş listeye sağ tıklayın ve **Disasm.'a kopyala (Copy to disassembler)**'yi seçin veya kod görünümünde **D**'ye basın. Bu, mevcut yordam için yeniden yapılandırılmış ifadeleri düşük seviye listede birleştirir. Konumunuzu korumak için komutu çağırırken imleci ilgili `loc_` adresi üzerine yerleştirin.

Bu iş akışını kullanırken **Adres Ekle (Add Addresses)** seçeneğini geri derleme ayarları bölümünde etkin tutun. Disasm. (Disassembler) her zaman adresleri gösterir; Adres Ekle, Decompiler görünümündeki adres gösterimini denetler.



Değişkenleri yeniden adlandırma

VB5/6 P-Kodu ve Yerel Kodu için bir değişkene sağ tıklayın ve **Ad değiştir (Rename variable)**'yi seçin veya imleci değişken üzerine yerleştirip **N**'ye basın. Anlamlı bir analiz adı girin. Geri derleyici, isim çakışmaları ve ayrılmış anahtar sözcükleri kontrol eder. Bir yeniden adlandırmayı geri almak için **Adı geri al (Undo renaming)** veya **Ctrl+Shift+Z**'yi kullanın. Bu komut sıradan metin düzenleme işlemlerinin geri almasından bağımsızdır.

Yeniden adlandırma, kurtarılan listeyi analiz etmeyi kolaylaştırır; derleme tarafından kaldırılan bir ismi kurtarmaz. Analistin verdiği adları bulguları aktarırken notlar olarak kabul edin.

Çağrıları ve küresel değişken referanslarını bulma

Çağrıları bulmak için Çözüm Gezgini (Solution explorer)'de bir yordama sağ tıklayın ve **Referansları bul (Find references)**'yi seçin veya ağaçta o yordam seçiliyken **X**'ye basın. Sonuçlar, ona tanımlı çağrıları içeren yordamları listeler. Bir sonuca çift tıklayarak çağrıyı inceleyin. Bu iş akışı VB5/6 ve .NET'i destekler; .NET desteği 12.4 sürümünde eklendi.

Bir küresel değişkeni araştırmak için imleci kod görünümündeki belirleyicisi üzerine yerleştirin ve **X**'ye basın. Aynı tuş, ağaçta yordam referanslarını ve kodda küresel değişken referanslarını arar. Tanınan referansların listesi, ek dolaylı, dinamik olarak çözülen veya kod karartmayla gizlenmiş kullanımların olmadığını kanıtlamaz.

Bağlama duyarlı kısayollar

Odak	Kısayol	Eylem
Kod	N	İmleç altındaki değişkeni yeniden adlandır (VB5/6).
Kod	X	İmleç altındaki küresel değişkene referansları bul.
Kod	D	Mevcut geri derlenmiş kodu Disasm.'a kopyala.
Kod	F4	Geçerli satırdaki adrese kadar izle; Tracing özellikleri gerektirir.
Kod	Ctrl+Shift+Z	İmleç altındaki değişkenin yeniden adlandırılmasını geri al.
Ağaç	X	Seçili yordama referansları bul.
Ağaç	G	Bölüme git iletişim kutusunu aç.
Ağaç	Ctrl+C	Seçili nesnenin tam ağaç yolunu kopyala.
Ağaç	Space or Insert	Seçili öğeyi işaretle veya işaretsiz bırak.

Tek harfli kısayollar odaklanan görünüme bağlıdır ve giriş alanlarında normal yazmayı değiştirmez. Odak değiştirmek için **Alt+1**'yi kullanın. Ctrl+T ve Ctrl+F için String Arama, ağaç yolları ve adresler için Adrese git ve emülatör sınırlamaları için Tracing'a bakın.

5 Yapay zekâ ile kod iyileştirme

YZ yardım (AI Helper), VB Decompiler tarafından zaten yeniden oluşturulmuş kodu işler. .NET assembly'lerinden geri derlenmiş C# ve P-Code veya Native Code'dan Visual Basic 5.0/6.0 kodlarını destekler. Yerel bir modeli Ollama üzerinden kullanabilir veya OpenAI API entegrasyonu aracılığıyla bir servise bağlanabilirsiniz.

Yapay zeka daha net kontrol akışı, değişken isimleri, tanımlamalar ve ifadeler önerebilir. Ancak ikili dosyada olmayan bilgileri güvenilir şekilde geri getiremez veya geri derleyicinin yanlış yorumladığı bir parçayı onaramaz. Geri kazanılmış kaynakta veya bir araştırma raporunda kullanmadan önce önerileri orijinal geri derlenmiş liste, makine kodu çözümlemesi ve çevre yordamlarle karşılaştırın.

Bir yordamı işleme

1. Gerekli yordamı Çözüm Gezgini (Solution explorer)'da açın ve **Decompiler** listesini inceleyin.
2. **YZ yardım**'ye geçin ve aşağıda belirtildiği gibi sağlayıcı, host ve modeli yapılandırın.
3. İşlemi başlatmak için model seçici yanındaki işlem düğmesini kullanın.
4. Dönen kodu YZ yardım'da inceleyin ve geri derleyici çıktısıyla karşılaştırın.

Yapay zeka işleme ayrı bir iş parçacığında çalışır. Model çok uzun süre yanıt veriyorsa **İptal (Cancel)**'ü kullanın. Ek bağlantı ayarlarını daraltarak kod için daha fazla alan bırakabilirsiniz.

Ollama üzerinden yerel bir model kullanma

1. resmi indirme sayfasından Ollama'yı yükleyin ve başlatın.
2. Bir terminalde `ollama pull qwen3:8b` komutunu çalıştırarak önerilen modeli indirin.
3. YZ yardım'da yerel LLM/Ollama sağlayıcısını seçin. **Ollama ana (Ollama host)**'yi sunucu adresine ayarlayın; varsayılan yerel adres `http://localhost:11434`'dir.
4. **Model**'de kurulu modeli seçin ve işlemi başlatın.

`qwen3:8b` modelini öneriyoruz. YZ yardım'ın yerleşik istemleri bu hafif modele göre uyarlanmıştır. Daha yeni modeller, sonucu iyileştirmeden kodu daha yavaş işleyebilir.

Model seçimi kısıtlanmamıştır: seçilen Ollama örneğinde kurulu herhangi bir model kullanılabilir. Yanıt biçimi tanınmıyorsa yanıt ayrıştırıcısı, modele özgü etiketleri veya diğer yardımcı metinleri görüntülenen sonuçta bırakabilir.

Ollama ile yerel çalıştırma için donanım

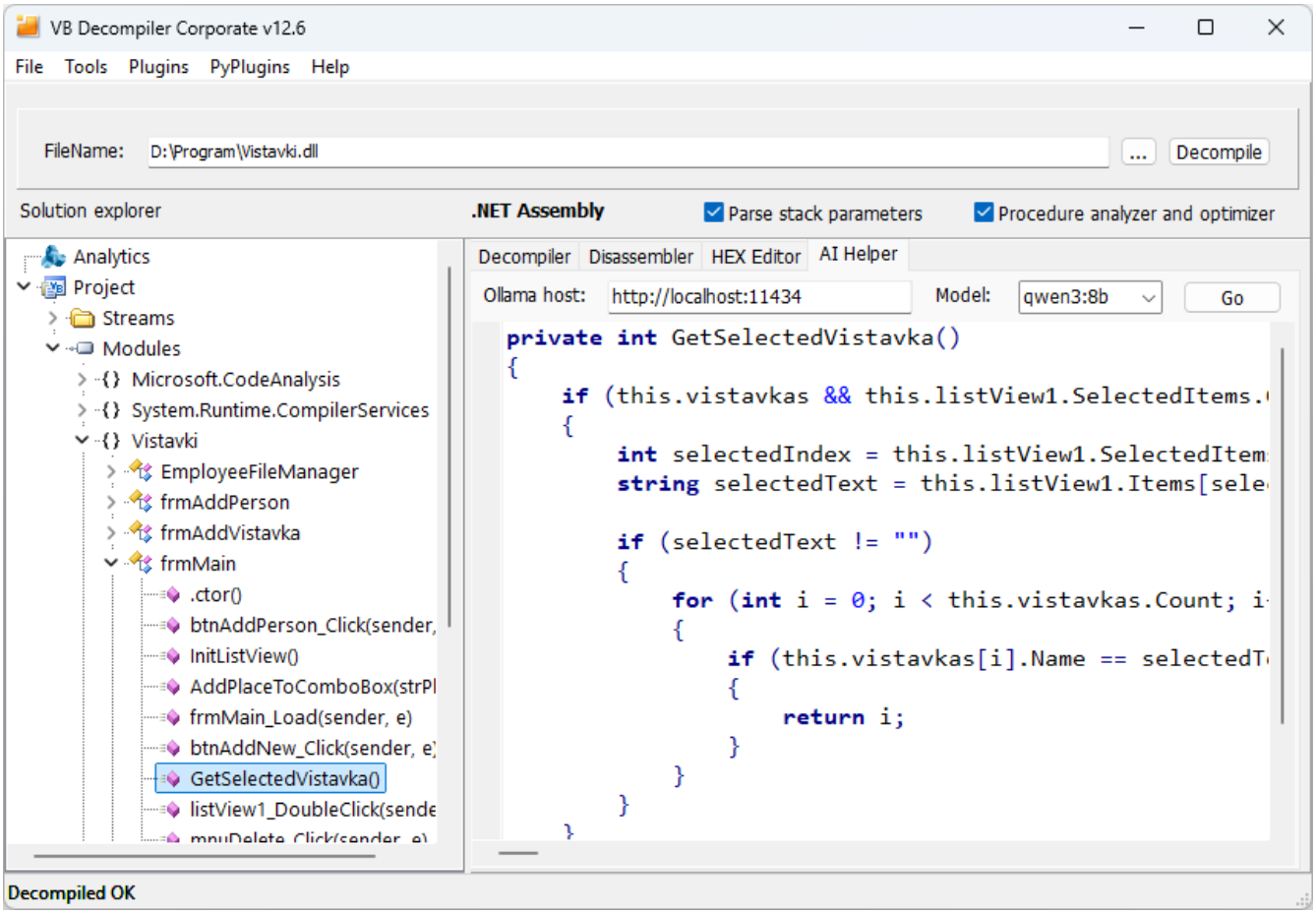
Yerel yapay zekâ işlemleri Ollama'nın çalıştığı bilgisayarda yürütülür ve geri derlemeden çok daha fazla kaynak gerektirir. Aşağıdaki pratik yapılandırmalar, `Q4_K_M` ağırlıkları ve standart `f16` bağlam önbelleği kullanan `qwen3:8b` modeliyle aynı anda bir isteğin işlenmesini temel alır. Bağlam token cinsinden ölçülür: 4K, 4096 token; 128K, 131072 token anlamına gelir. Bu sınıra talimatlar, kaynak kodu ve üretilen yanıt dahildir.

Bileşen	Pratik asgari yapılandırma: 4K	Önerilen yapılandırma: 128K
İşletim sistemi	64 bit Windows 10 (22H2) veya Windows 11	64 bit Windows 10 (22H2) veya Windows 11
İşlemci	AVX2 destekli, en az 4 fiziksel çekirdekli modern x64 CPU	AVX2 destekli, en az 8 fiziksel çekirdekli modern x64 CPU
Sistem belleği (RAM)	16 GB	64 GB
Ekran kartı	En az 8 GB ayrılmış VRAM içeren, Ollama uyumlu GPU	En az 32 GB ayrılmış VRAM içeren, Ollama uyumlu GPU
Depolama	Ollama ve bu model için en az 20 GB boş alanı olan SSD	Ollama ve bu model için en az 20 GB boş alanı olan SSD

Bağlam boyutunu Ollama ayarlarındaki kaydırıcıyla seçin. 4K için uygun olan 8 GB ekran kartı, yukarıdaki 128K yapılandırması için yeterli değildir. Model indirmesi yaklaşık 5.2 GB boyutundadır; toplam bellek gereksinimi daha yüksektir. 128K bağlamda yalnızca `f16` önbelleği yaklaşık 18 GiB kullanır; model ağırlıkları ve çalışma arabellekleri için de bellek gerekir. Windows ve diğer uygulamalar için boş bellek bırakın.

Yeterli RAM varsa Ollama CPU üzerinde çalışabilir; ancak kod işleme çok yavaş olabilir. Pentium M sistemler ve benzer yaştaki bilgisayarlar, `qwen3:8b` modelini yerel çalıştırmaya uygun değildir. Eski Intel HD Graphics birimlerinin paylaşılan belleği, yukarıda belirtilen ayrılmış VRAM yerine geçmez. Bu gereksinimler Ollama'nın çalıştığı bilgisayar içindir; uzak sunucular ve harici API hizmetleri kendi donanımlarını kullanır.

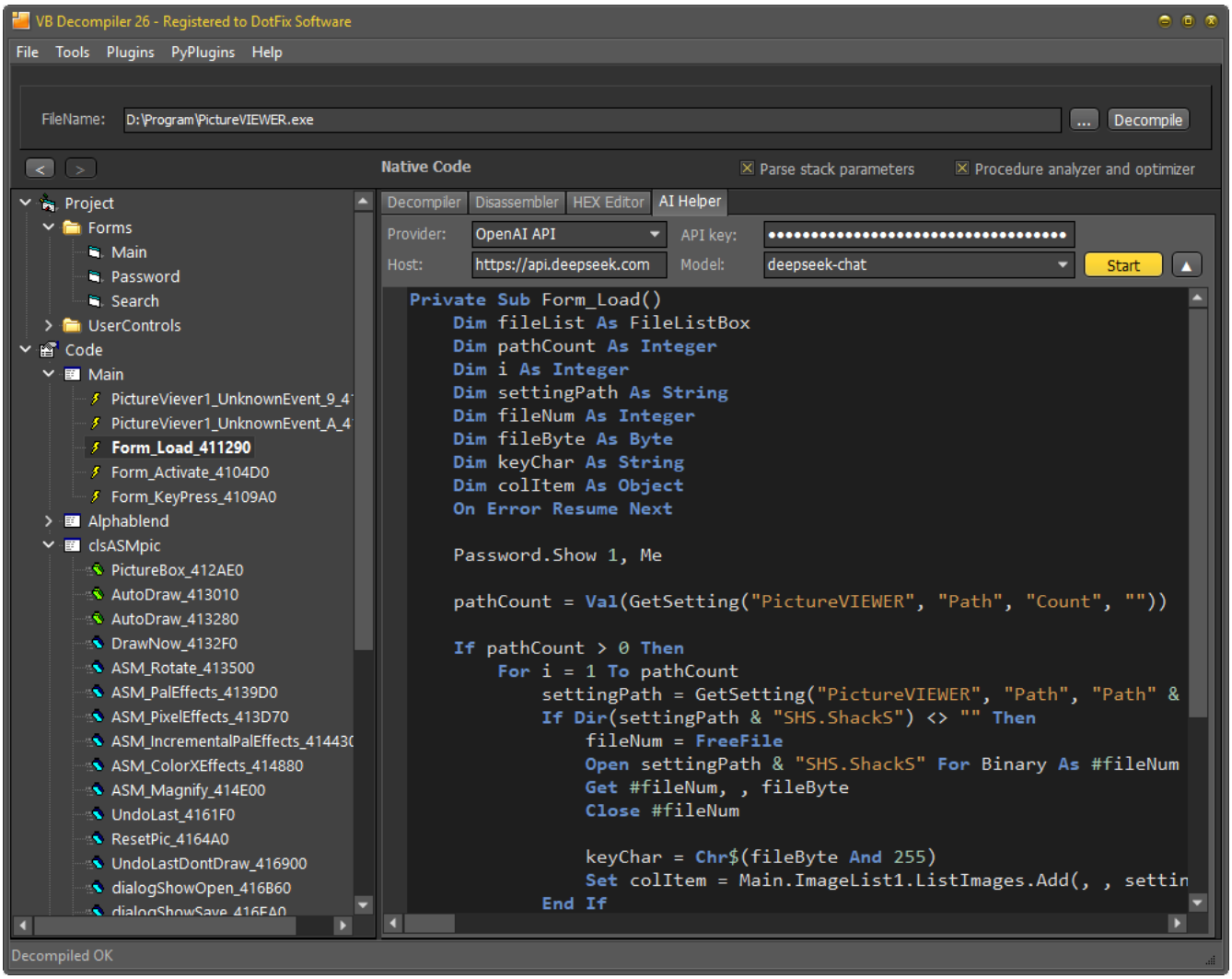
Kısa yordamlar için 4K ile başlayın. Daha büyük bağlam, bellek kullanımını ve uzun girdilerin işlenme süresini artırır. İşlem çok yavaşsa iptal edin, bağlamı küçültün ve `ollama ps` ile GPU kullanımını kontrol edin. Gerçek bellek kullanımı Ollama sürümüne, hesaplama arka ucuna ve çalışan diğer görevlere bağlıdır.



OpenAI API entegrasyonu kullanma

Servis (Provider)'i **OpenAI API**'ye ayarlayın. **Host**, **API key** ve **Model**'yi hedeflenen servis için yapılandırın. Geçerli bir API anahtarı ve bu anahtarla kullanılabilir bir model tanımlayıcısı girin. API anahtarlarını ekran görüntülerinden ve paylaşılan analiz materyallerinden çıkarın.

YZ yardım ayrıca **OpenAI API** sağlayıcısı üzerinden DeepSeek'i destekler. DeepSeek hostunu, API anahtarını ve model tanımlayıcısını ilgili alanlara girin. Aşağıdaki ekran görüntüsü bu yapılandırmayı gösterir.



Kodun işlendiği yer

YZ yardım, mevcut geri derlenmiş kodu yapılandırılmış servise gönderir. Sadece yerel bir iş akışı için yerel olarak çalışan bir Ollama sunucusu ve yerel olarak yürütülen bir model kullanın. Başka bir makinede bulunan bir Ollama hostu, Ollama bulut modeli veya dış API servisi kodu analiz istasyonundan dışarı aktarabilir.

Gizli kaynak kod kurtarma veya güvenlik araştırmaları için, organizasyonunuzun veri işleme gereksinimlerine göre uç noktayı seçin. Sadece Ollama'nın varlığından veya localhost URL'sinden offline çalışma çıkarımı yapmayın.

Sorun giderme

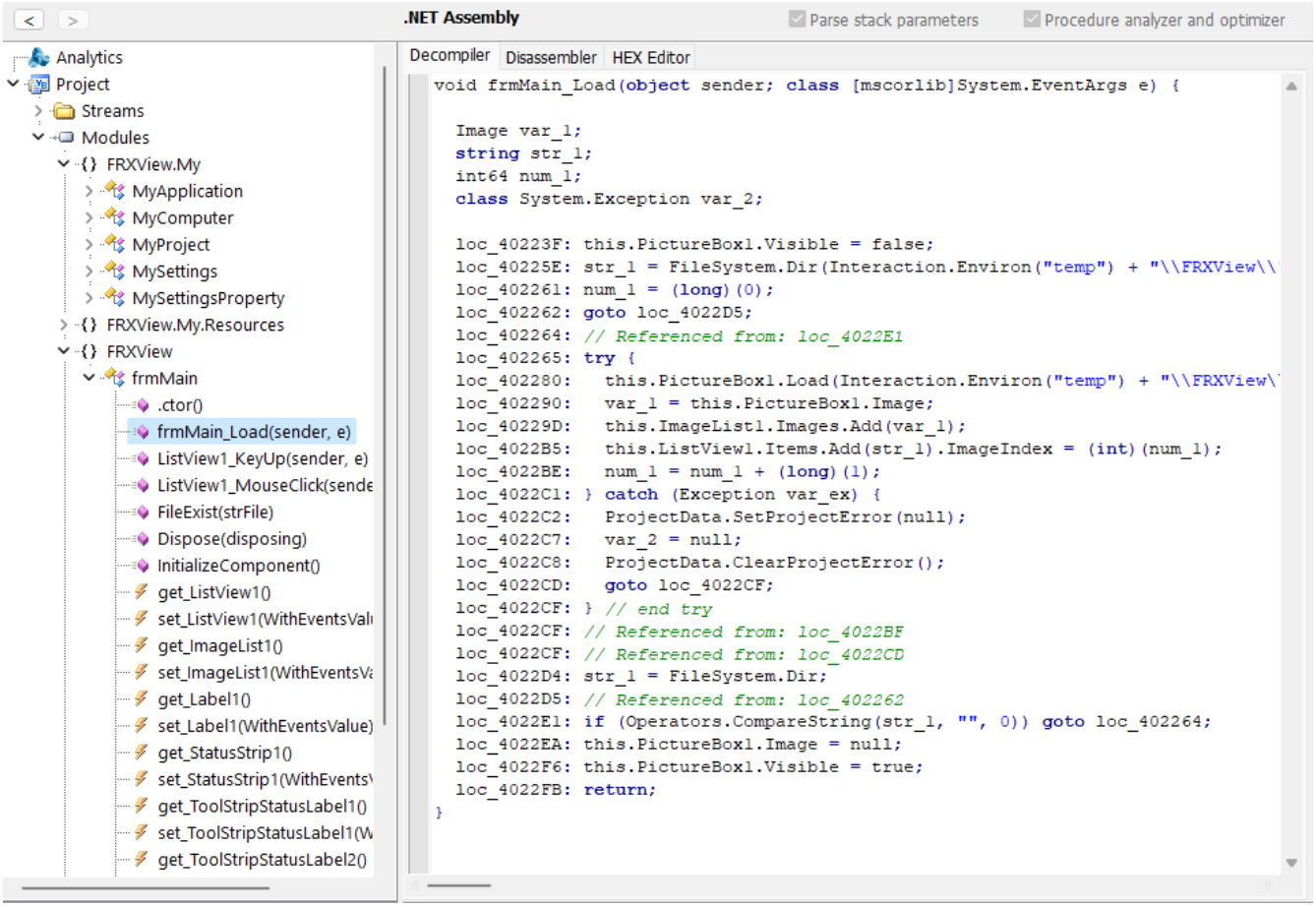
- Eğer yerel bir model eksikse, host alanı tarafından belirtilen Ollama örneğinde kurulu olduğundan emin olun.
- Eğer yanıt yoksa, seçilen servisin çalıştığına ve erişilebilir olduğuna ve uç nokta, kimlik bilgisi ve model tanımlayıcısının aynı sağlayıcıya karşılık geldiğine bakın.
- Eğer işleme çok yavaşsa iptal edin. Başka bir yerel model kullanırken önerilen `qwen3:8b` modelini deneyin.
- Eğer tanımadığınız bir model yanıtında etiketler veya diğer yardımcı metin bırakırsa, `qwen3:8b`'yi deneyin veya sonucu gözden geçirirken bu metni manuel olarak kaldırın.
- Eğer sonuç beklenmedik şekilde bir koşulu, çağrıyı, sabiti veya türü değiştirirse, düşük seviye liste ile doğrulayın. Okunabilirlik tek başına eşdeğer davranışı kanıtlamaz.

Ollama ve dış model sağlayıcıları üçüncü taraf ürünlerdir. Kullanılabilirlikleri, model davranışları ve servis politikaları VB Decompiler'dan bağımsızdır.

6 Açma ve kaydetme

6.1 Dosya açma

Geri derleme işlemine başlamadan önce analiz edilecek dosyayı seçmeniz gerekir. Bunun için **Programı Aç (Open program)** seçeneğini **Dosya (File)** menüsünden kullanın. **VB Decompiler**, **Visual Basic** 5.0 veya 6.0 ile oluşturulmuş EXE/OCX/DLL dosyalarının yanı sıra **Visual Studio .NET** ile derlenmiş uygulamaları da açabilir. İkinci durumda kaynak dil önemli değildir; Visual Studio .NET, uygulamaları kaynak dilden bağımsız olarak ortak bir iç biçime, **MS IL** biçimine derler.



Dosya açıldığında geri derleyici, programın belirtilen geliştirme ortamlarından biriyle oluşturulup oluşturulmadığını denetler. Program desteklenmeyen bir ortamda (örneğin **Delphi** veya **Visual C++**) geliştirilmişse, ortamın desteklenmediği ve programın geri derlenemeyeceği bildirilir. Bu mesajı görürseniz bizimle iletişime geçebilirsiniz; programınızın geri derlenmesine yardımcı olabiliriz. **Delphi** ve **FreePascal** uygulamalarının geri derlenmesi için dahili araçlarımız vardır ve bu dosyalar için ayrı bir geri derleme hizmeti sunabiliriz.

Açılan program bir **EXE paketleyicisi** veya kod karartma aracıyla paketlenmiş ya da korunmuşsa, geri derleyici paketi açmayı denemeyi önerir. Bu işlev yalnızca 32 bit Windows sürümlerinde kullanılabilir ve herhangi bir garanti verilmez. Paket açılabiliyorsa dosya çalıştırılır, bellekte açılır, farklı bir adla diske kaydedilir ve geri derleme için açılır. **Dikkat!** Yerleşik paket açıcıyı kötü amaçlı yazılımlar veya sisteme zarar verebilecek başka kodlar üzerinde asla kullanmayın. Paket açma sırasında hedef program çalıştırılır ve bilgisayara ciddi zarar verebilir.

Önemli not: Gerekli haklara sahip olmadan veya hak sahibinin lisansla dosyanın paketinin açılmasını ve analizini yasakladığı durumlarda yazılımı geri derlemeyi ya da paketini açmayı denemeyin. Bu, ülkenizin yasalarını ihlal edebilir.

*Visual Studio .NET, Visual C++ ve Visual Basic, Microsoft Corporation'ın tescilli ticari markalarıdır. Delphi, Embarcadero'nun tescilli ticari markasıdır.

Diğer dosya açma yöntemleri

Bir dosyayı Windows Gezgini'nden VB Decompiler penceresine sürükleyebilirsiniz. Bilinmeyen uzantılı veya uzantısız dosyalar da bu şekilde açılabilir; tanıma işlemi dosyanın gerçek içeriğine bağlıdır. Desteklenen Windows **.lnk** kısayolları hedef dosyalarına çözümlenebilir. Kayıtlı VDD veritabanları da sürükleyip bırakarak veya komut satırından açılabilir.

6.2 Analiz veritabanlarıyla çalışma

Bu özellik, **Veritabanı (Data Base)** bölümü üzerinden **Dosya (File)** menüsünde kullanılabilir ve geri derlenmiş sonuçların ayrı bir sıkıştırılmış dosyaya kaydedilmesini sağlar. Bu özellik, gelecekte geri derlenmiş sonuçların kodu yeniden geri derlemeye gerek kalmadan daha hızlı yüklenmesini sağlayabilir.

Eğer programı tekrar analiz etmeniz veya bu sonuçları bir meslektaşınızla paylaşmanız gerekiyorsa, geri derleme işlemi tekrar yapmanıza gerek yoktur. Tam geri derleme (**Hızlı Geri Derleme (Fast decompilation)** seçeneği devre dışı bırakıldığında) bu işlem daha uzun sürebilir ve ek kaynak gerektirebilir. Geri derlenmiş kodu önceden **Database**'e kaydetmek çok daha uygundur.

VB Decompiler, içsel sıkıştırılmış bir veritabanı formatı kullanır, bu da böyle bir veritabanında bulunan saklanan bilgilerin kapladığı alanın azaltılmasını sağlar. Ayrıca, VB Decompiler'ın çoğu önceki sürümü tarafından kaydedilen veritabanlarıyla uyumluluk bulunur, böylece ürünün yeni sürümünde böyle bir veritabanını açamama riski olmadan bilgiyi saklayabilirsiniz.

VDD veritabanlarının açılması ve aktarılması

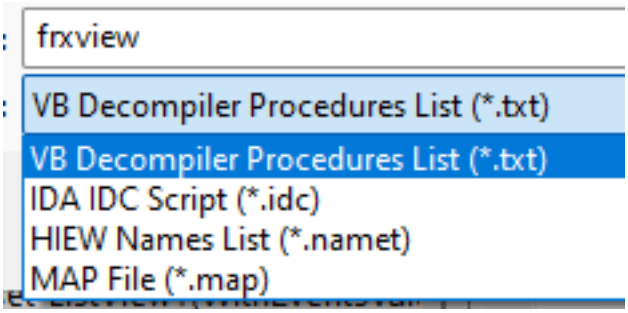
Kaydedilmiş analiz veritabanları **.vdd** uzantısını kullanır. VDD veritabanını dosya açma iş akışı üzerinden, geri derleyici penceresine sürükleyerek veya yolunu komut satırında belirterek açabilirsiniz.

Orijinal analiz edilen ikiliyi bulduğunuzda, VB Decompiler hem kaydedilmiş yolunu hem de veritabanı bulunan dizini kontrol eder. İşbirliği için, orijinal yolu kullanılamadığında VDD dosyasının yanına karşılık gelen ikilinin bulunmasını sağlayın. Adreslerin ve dosya baytlarının anlamlı kalması için tam analiz edilen sürümü koruyun.

6.3 Yordam listesini kaydetme

Bu işlem, **Prosedür listesini kaydet (Save procedures list)** seçeneğini **Dosya (File)** menüsünde çağırarak gerçekleştirilir ve programları analiz etmede güçlü bir araçtır; çünkü dış yardımcı programların **VB Decompiler**'i desteklemesine olanak tanır.

Şu anda, işlenen programda kullanılan adresler, fonksiyon adları ve yordamlar birkaç formatta kaydedilebilir. Standart kaydetme iletişim kutusunda dosya türünü belirterek bu seçeneklerden herhangi birini seçebilirsiniz.



TXT biçiminde kaydetme, yordam listesini basit bir metin dosyasında kaydetmenizi sağlar. Bu, programların farklı versiyonları arasındaki yordamları karşılaştırmak ve adres değişikliklerini belirlemek gibi işlemler için yararlı olabilir.

IDC betiği olarak kaydetme, fonksiyon ve yordam adreslerini **IDA Pro** disassembler'ına yüklemenizi sağlar. Analiz edilen programın **Native Code**'ye derlenmesi durumunda bu özellik oldukça yararlı olabilir.

NAMET biçiminde kaydetme, **HIEW** yardımcı programını kullanarak analizi basitleştirmenizi sağlar. Elde edilen dosyayı analiz edilen programın klasörüne kopyalayın. Ardından programı HIEW'de açın ve F12 tuşuna basın. Boş yordam listesini içeren bir pencere görünecektir. F12 tuşuna yeniden basın. NAMET dosyasının yolu oluşturduğunuz dosyayla uyuşuyorsa Enter tuşuna basın. Yordam adresleri HIEW'e yüklenecek ve F12 tuşuna basarak yöntemler arasında gezinebileceksiniz.

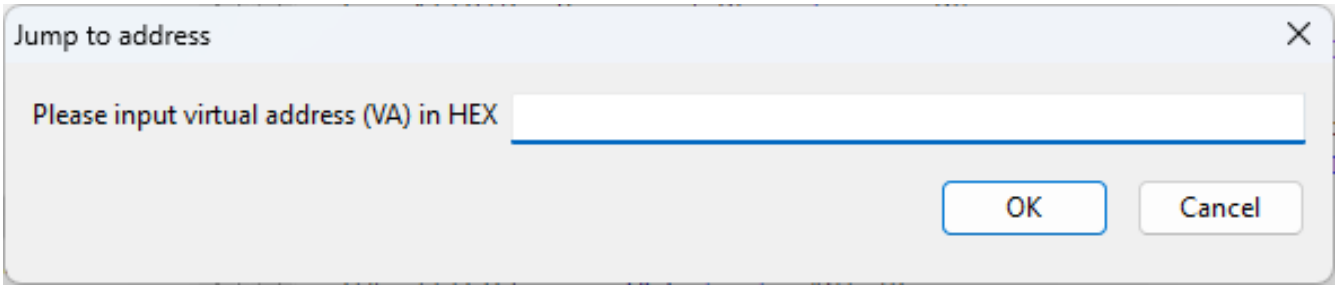
MAP biçiminde kaydetme, derleyiciler tarafından genellikle oluşturulan standart harita dosyası oluşturur. Bu dosya, program fonksiyonlarının başlangıcını işaretlemek için pek çok hata ayıklayıcı ve disassembler ile kullanılabilir. Visual Basic ile yazılan programların makine kodunun üçüncü taraf araçlarda yanlış çözümlenebilmesi nedeniyle, **VB Decompiler**'de oluşturulan MAP dosyası süreci büyük ölçüde basitleştirecektir. Bu format, **OllyDbg**, **IDA Pro** ve diğerleri gibi çeşitli ürünlerde eklentiler aracılığıyla desteklenmektedir.

7 Program işlevleri

7.1 Adrese gitme

Sanal adrese gitme

Araçlar > Sanal adrese geç (Tools > Jump to virtual address)'yi seçin veya **Ctrl+G**'ye basın. İncelemek istediğiniz kodun sanal adresini girin. Ayrıca, listede bir adresi seçip Ctrl+G'yi de kullanabilirsiniz. Geri derleyici, o adresle ilişkili kodu açar.



Sanal adres (VA), yürütülebilir görüntünün adres alanındaki bir konumu tanımlar ve görüntü tabanını içerir. Dosya ofseti, diskteki dosyadaki bayt pozisyonunu belirler. Biri yerine diğerini kullanmayın: HEX Düzenleyici her iki yönlendirme biçimini de destekler.

Gezinme geçmişinden geri dönme

Önceden görüntülenen yordamlar arasında ileri ve geri hareket etmek için ağaç üzerindeki ok düğmelerini kullanın. **Esc**, önceki yordama döner. Mouse Geri ve İleri düğmeleri de bu geçmişi kullanarak, yordam içindeki pozisyon korunur.

Ağaç yoluyla bir konumu paylaşma

Solution explorersinde, gerekli nesneye sağ tıklayın ve **Bölümün adını kopyala (Copy the name of section)**'ü seçin veya ağaç odakta iken **Ctrl+C**'ye basın. Bu, seçili düğümün tam yolunu kopyalar.

O konuma geri dönmek için ağaç bağlam menüsünde **Bölüme geç (Jump to the section)**'yi seçin veya ağaçta **G**'ye basın ve kopyalanan yolu iletişim kutusuna yapıştırın. Bu, bir soruşturmayı başka bir analiste iletirken veya büyük bir projeye geri dönerken kullanışlıdır. Yolu ilgili analiz edilen dosya ile kullanın; başka bir derlemeye ait yol artık aynı kodu tanımlaymayabilir.

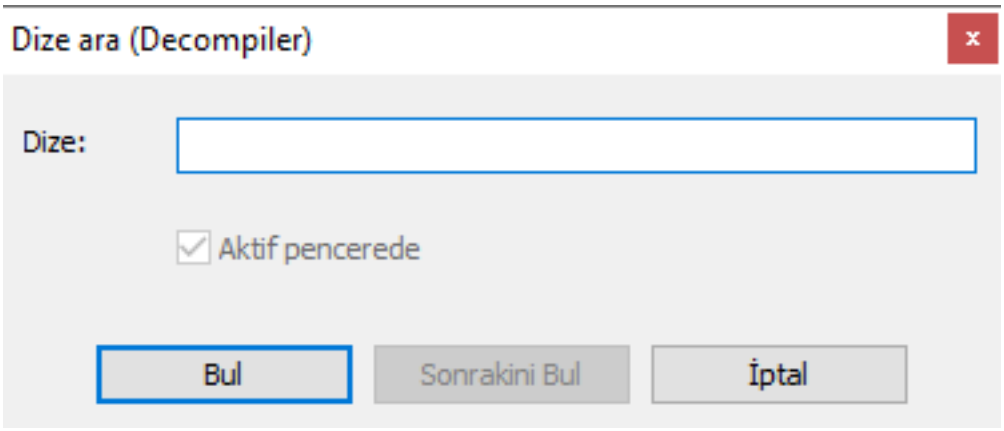
7.2 Dize arama

İsme göre bir yordam bulma

Araçlar > Yordam bul (Tools > Find procedure)'yi seçin veya **Ctrl+T**'ye basın. Tam bir yordam adı veya onun belirleyici bir kısmını girin. Arama proje ağacını kapsadığından, yordamın hangi modülde veya sınıfta bulunduğunu bilmenize gerek yoktur. Birden fazla isim eşleşirse, sonuç listesi tüm eşleşen girişleri içerir; gerekli girişi açmak için çift tıklayın.

Görüntülenen metin veya tüm projede arama

Araçlar > Find string (Tools > Find string)'yi seçin veya **Ctrl+F**'ye basın ve metin arama iletişim kutusunu açın. **Aktif pencerede (In active window)** seçiliyken, arama geçerli olarak görüntülenen kod veya form verilerine uygulanır. Bu onay işaretini temizleyerek geri derlenmiş projede arama yapabilirsiniz.



Dize (String)'de metni girin ve **Bul (Find)**'yi seçin. Aynı metni aramaya devam etmek için **Sonrakini Bul (Find next)**'yi kullanın. **İptal (Cancel)** iletişim kutusunu kapatır.

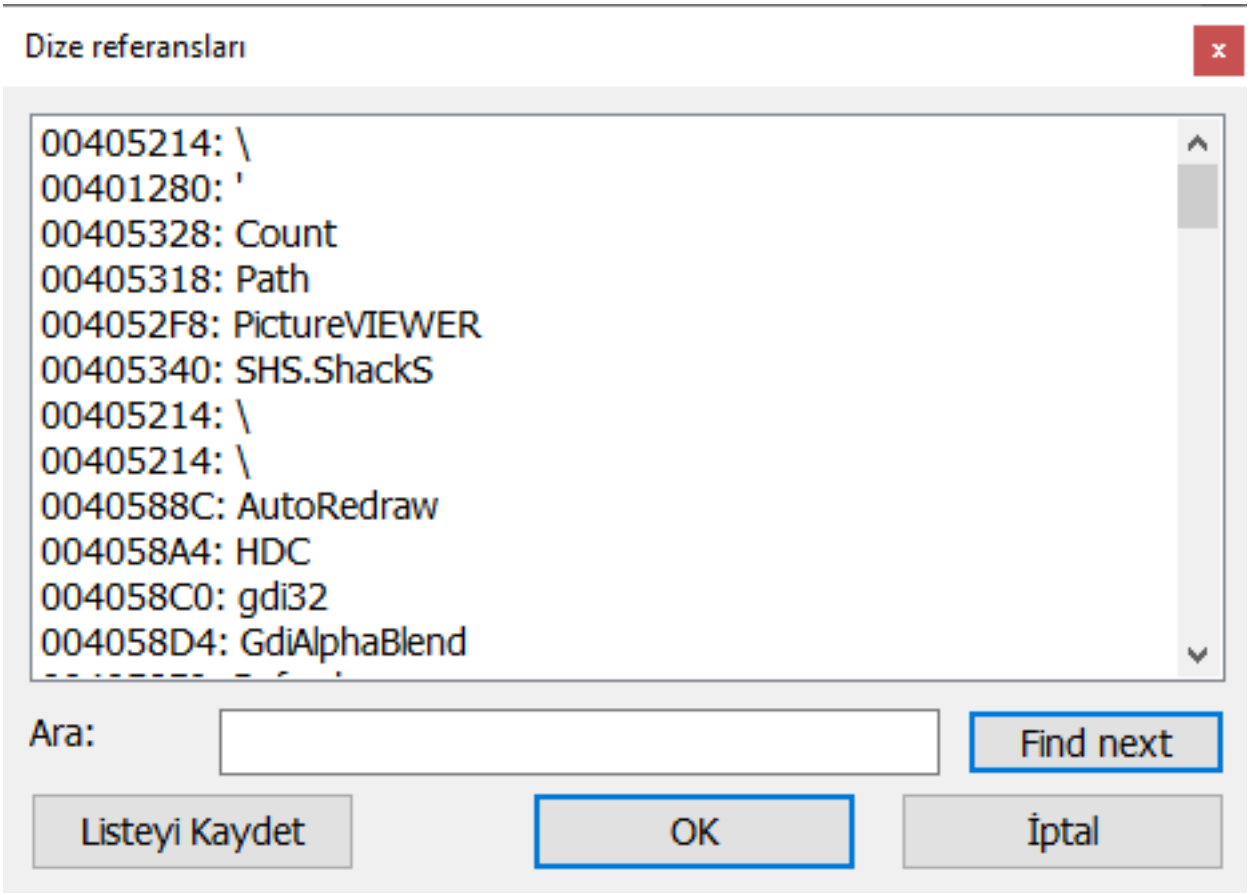
Tüm geri derlenmiş metni aramak için tam geri derleme gerekir. Önce **Hızlı Geri Derleme (Fast decompilation)** seçeneğini Gelişmiş özellikler bölümünde devre dışı bırakın, ardından dosyayı yeniden işleyin. Yordamları adlarına göre aramak, proje yapısı üzerinde gerçekleştirilen ayrı bir işlemdir.

Dize referanslarını arama

Eğer dize sabitlerine ve bunları kullanan koda ihtiyacınız varsa, Dize Referansları'nı açın. **Ara (Search)** alanı ve **Sonrakini Bul** düğmesi dize listesini arar. Dize referansları, Hızlı Geri derleme modunda ön makine kodu çözümlemesi ile toplanabilir; her tam geri derlenmiş yordamın aranmasını gerektirmezler.

7.3 Dize başvuruları

Herhangi bir programın kodunda, genellikle dize verileri (kayıt anahtarları, dosya adları, hata mesajları vb.) ile karşılaşırsınız. Geri derlenmiş uygulamalarda kullanılan yordam ve fonksiyonların kullandığı dize listesini görmek oldukça yararlı olabilir. Bu tam olarak **Dize referansları (String references)** fonksiyonu için tasarlanmıştır, **Araçlar (Tools)** menüsünden erişilebilir. Kullanmak için, **Araçlar** -> **Dize referansları**'ye gidin ve programınız herhangi bir dize verisi içeriyorsa, aşağıda gösterildiği gibi görüntülenecekler:

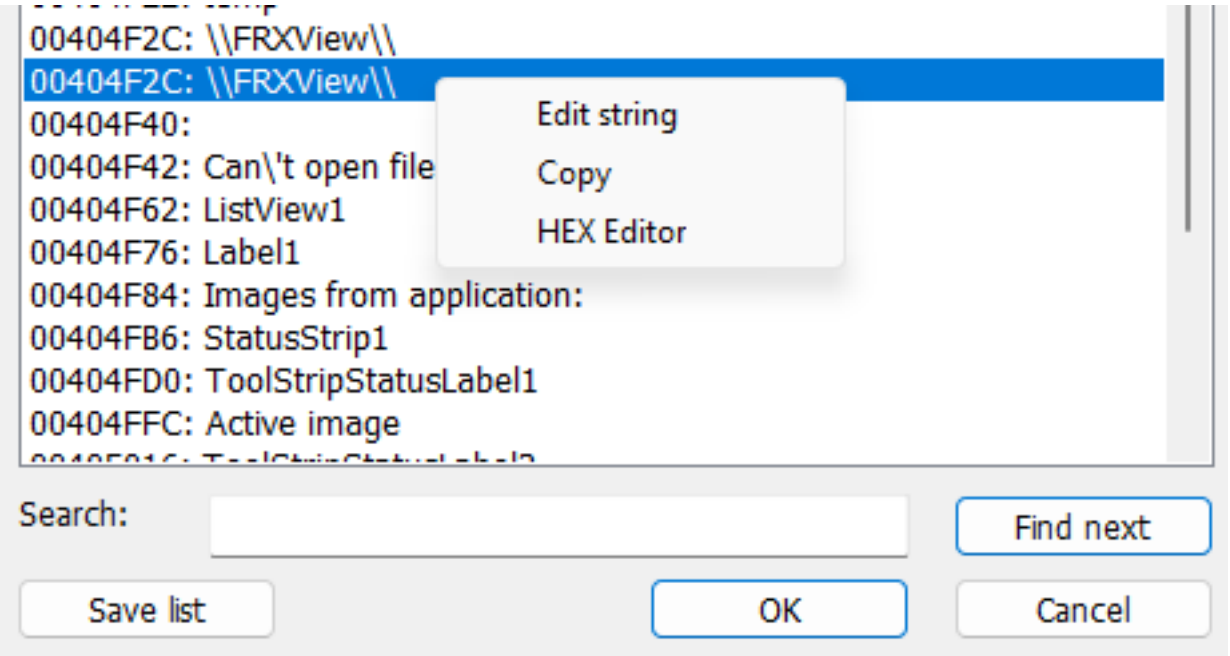


Listede istediğiniz satırı bulun ve üzerine çift tıklayın. O satırı içeren kod parçası bulunacak ve ilgili yordam kod penceresinde açılacaktır. Satıra tekrar çift tıklamak, bu satırın programın kodu içinde aramasını sürdürür. 'Tamam'a tıklamak, kodda vurgulanmış satırı bulur ve satır seçim iletişim kutusunu kapatır. İptal etmek sadece diyalog kutusunu kapatır.

Dize referansları fonksiyonu yalnızca yazılımın lisanslı sürümünde mevcuttur ve **Lite** sürümü için erişilemez.

Önemli not: dizeler, 1.x programındaki gibi yalnızca aktif modül için değil, tüm program için gösterilir. Bu değişiklik, büyük sayıda form ve modüle sahip programların analizini kolaylaştırmak amacıyla yapılmıştır.

VB Decompiler ayrıca dizeleri düzenlemeyi destekler. Dizeyi değiştirmek istediğiniz dizeye sağ tıklayın ve oluşan bağlam menüsünden **Dizeyi düzenle (Edit string)**'yi seçin. Yeni bir dize girmek için bir iletişim kutusu belirecektir. Dizeyi değiştirdikten sonra, **Ok**'ye tıklayın ve dize doğrudan düzenlenmiş dosyada değiştirilecektir. Bu işlem sırasında yedek alınmadığından ve bu işlemin geri alınamayacağından, incelenen dosyanın yedeğini önceden oluşturmanız önerilir.



Önemli not: dizeleri düzenlerken, yeni dizenin uzunluğunun eski olanın uzunluğunu aşmamasına dikkat edin. Bu, daha uzun bir dize eklemek için boşluk ayrılmayan ikili uygulama dosyalarının formatıyla ilgilidir.

Dizelerin listesini içeren pencerede, HEX Düzenleyici menüsünden seçerek dizeyi HEX düzenleyicide görüntüleyebilir ve kısmen düzenleyebilirsiniz. HEX düzenleyicide, değişiklikleri geri alma ve yeniden yapma işlevleri vardır, bu da uygulama dize verilerini değiştirme sırasında denemeyi kolaylaştırır.

Dize listesi penceresi de **Ara (Search)** adında bir metin alanına ve **Sonrakini Bul (Find next)** düğmesine sahiptir. Bu işlevsellik, özellikle büyük projelerle uğraşırken manuel görsel arama ile listeyi geçmek son derece zor olduğunda, liste içindeki dizeleri bulmak için yararlı olacaktır.

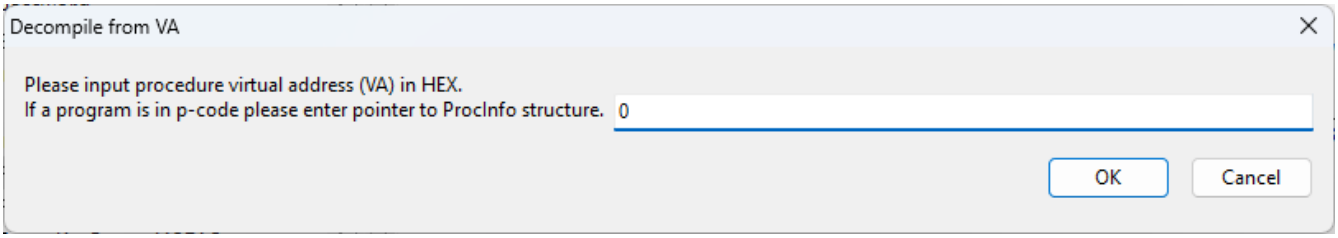
Hızlı Geri derleme ve büyük dize listeleri

Dize referansları, ön makine kodu çözümlemesi dahil **Hızlı Geri Derleme (Fast decompilation)** modunda toplanabilir. Bu, tüm geri derlenmiş metni aramaktan farklıdır. 26.2 sürümü, 32,000'den fazla referans içeren listeleri destekler ve büyük listeler için bellek yönetimini iyileştirir.

.NET derlemeleri için, #US öbeğinden kullanıcı dizeleri dosya konumlarıyla birlikte incelenebilir. Liste, statik analiz için kullanılabilir dizeleri temsil eder; şifreli veya dinamik olarak oluşturulan dizeler ayrı inceleme gerektirebilir.

7.4 Belirtilen adresten geri derleme

Araçlar (Tools) menüsünde, çok kullanışlı bir fonksiyon bulunmaktadır - **Adresten Geri Derle (Decompile from VA)** (verilen sanal adresten herhangi bir kodu geri derleme). Bu fonksiyonun temel özelliği, geri derlenecek dosyanın doğru yapıya sahip olması ve standart yöntemle işlenmesi gerektirmemesidir. Tabii ki bu durumda kod, bazı sınırlamalarla geri derlenecektir. Örneğin, içe aktarılan işlevler veya OLE nesneleri önceden işlenmediyse, kod o noktalarda bilgilendirici olmayacaktır.



Bir dosya içinde belirli bir kod parçasını geri derlemek için, sadece o kodun sanal adresini belirtmeniz gerekir (eğer **Native Code** veya **.NET derlemesi**'den bahsediyorsak), ya da bir **ProcInfo** yapı referansı sağlamalısınız (eğer klasik **Visual Basic P-Code**'den bahsediyorsak). Böyle **P-Code** ile çalışmanın benzersizliği, **ProcInfo** yapısının kod tarafından kullanılan bir veri dizisini içermesidir. Bu dizisiz, kodun hangi fonksiyonlarla çalıştığını, hangi dizelere atıfta bulunduğunu vb. belirlemek imkansızdır. **.NET derlemeleri** ile çalışırken, adres girildikten sonra geri derleyici size kodu **MS IL** ara kod olarak geri derlemek veya makine kodunu çözümlemek arasında seçim yapmanızı teklif edecektir. Bu seçenek, .NET derlemelerinin hem MS IL ara kodunu (**yönetilen kod**) hem de makine kodunu (yani **yönetilmeyen kod** olarak adlandırılan) içerebilecekleri gerçeğine dayanmaktadır.

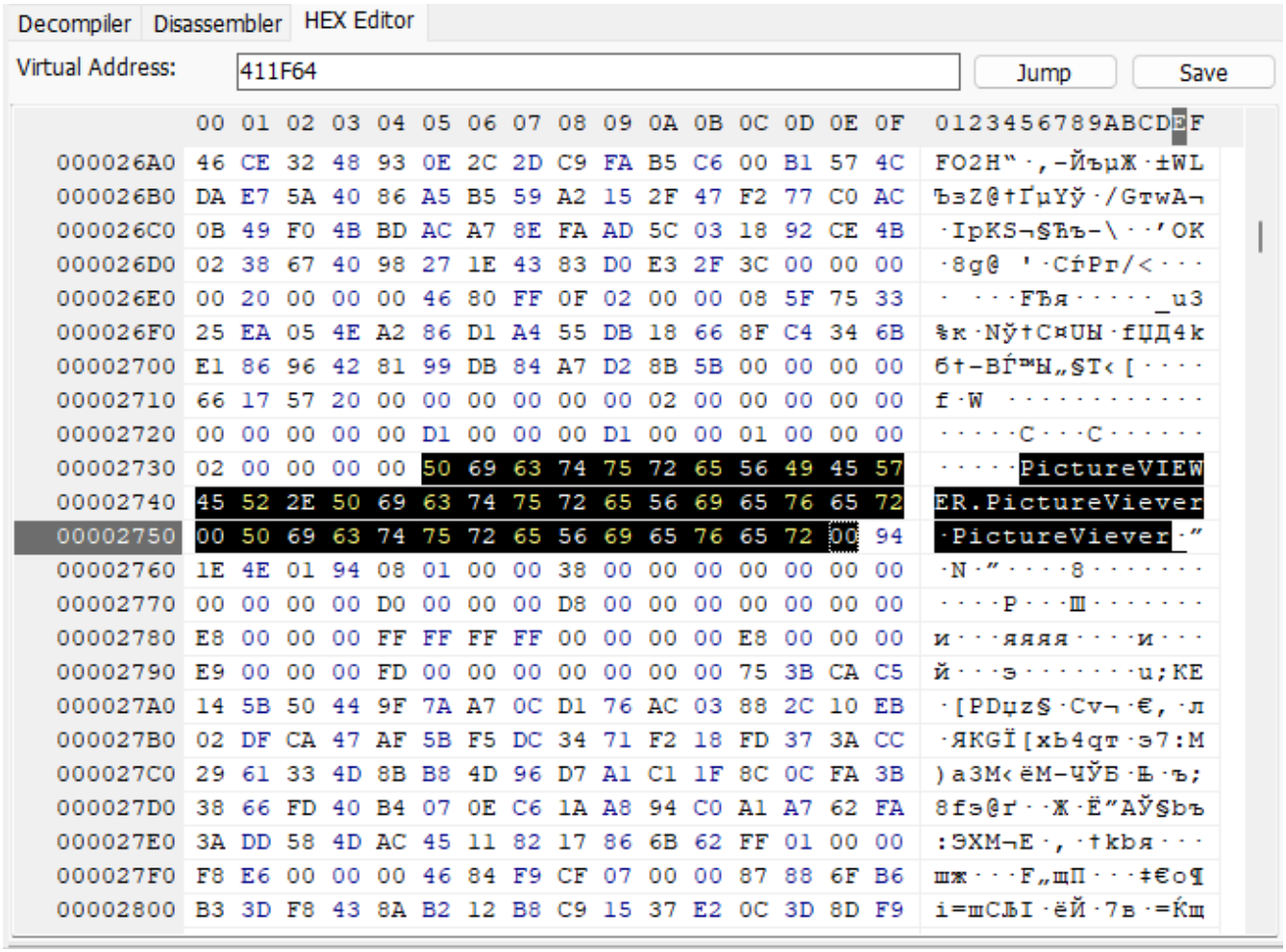
Dosya zaten geri derleyicide açılmışsa, sadece **Adresten Geri Derle** fonksiyonunu çağırmak ve işlenecek adresi belirtmeniz gerekir. Dosya henüz açılmadıysa, bu otomatik olarak önerilecek ve dosya basitleştirilmiş modda açılacaktır.

Adresten Geri Derle fonksiyonu, istenen kodun otomatik olarak geri derlenmediği veya dosyanın kısmen hasar görmüş olduğu ve standart araçlarla standart yöntemlerle işlenemediği durumlarda özellikle kullanışlıdır. Tabii ki bu durumda geri derleme sonucunun, standart geri derlemeden daha kötü olacağını anlamalısınız.

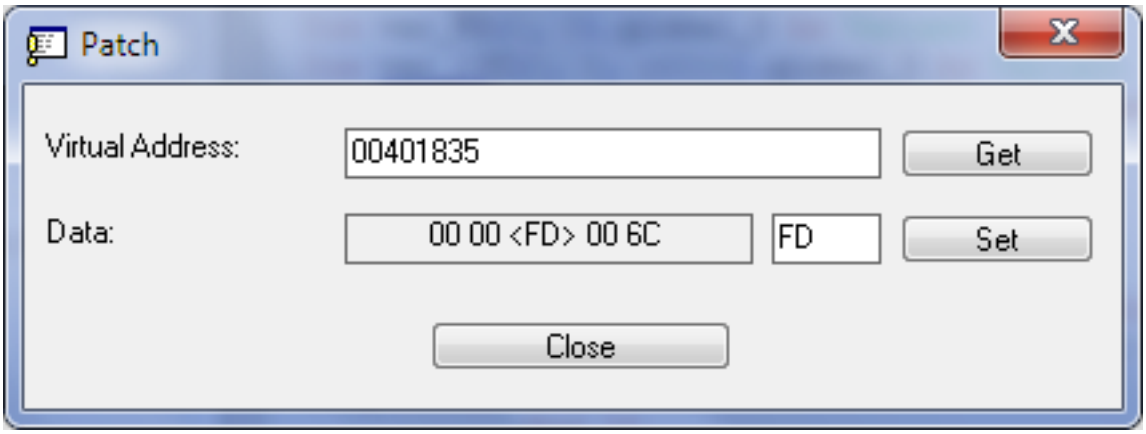
7.5 Verileri değiştirme

9.4 sürümünden itibaren, **VB Decompiler**, dosyalar içinde ikili veri düzenlemek için modern ve kullanışlı bir araçla birlikte gelir - tam donanımlı bir **HEX Editörü (HEX Editor)**. Bu araç, her yordam veya form için geri derlenmiş kodun yanındaki **HEX Editörü** sekmesinden ana program penceresinden erişilebilir.

HEX Editörü, dosyayı bayt bayt ve bloklar halinde düzenlemenize olanak tanır. İncelenen dosya içinde istenen adrese gitmek için sanal adresler (şunları da içerir: **Image Base**) veya ofsetler (dosyanın başlangıcına göre gerekli verinin ofsetini gösteren düzenli bir adres) kullanabilirsiniz.



Decompiler'ın daha önceki sürümlerinde (9.4 sürümünden önce), dosyaları düzenleme işlevi daha basit bir araç olan Data Patcher kullanılarak uygulanırdı. Çalışma şekli şöyledir: **Araçlar (Tools)** menüsü -> **Yama Verileri (Patch data)** altında, programın kodunu analiz ettikten sonra bazı kısımları değiştirmek gerektiğinde kullanılabilecek bir patcher aleti vardı. Gerekli baytları dış araç kullanmadan değiştirebilmek için **VB Decompiler**'nin yerleşik işlevselliği mevcuttu. Patcher penceresi şöyle görünüyordu:



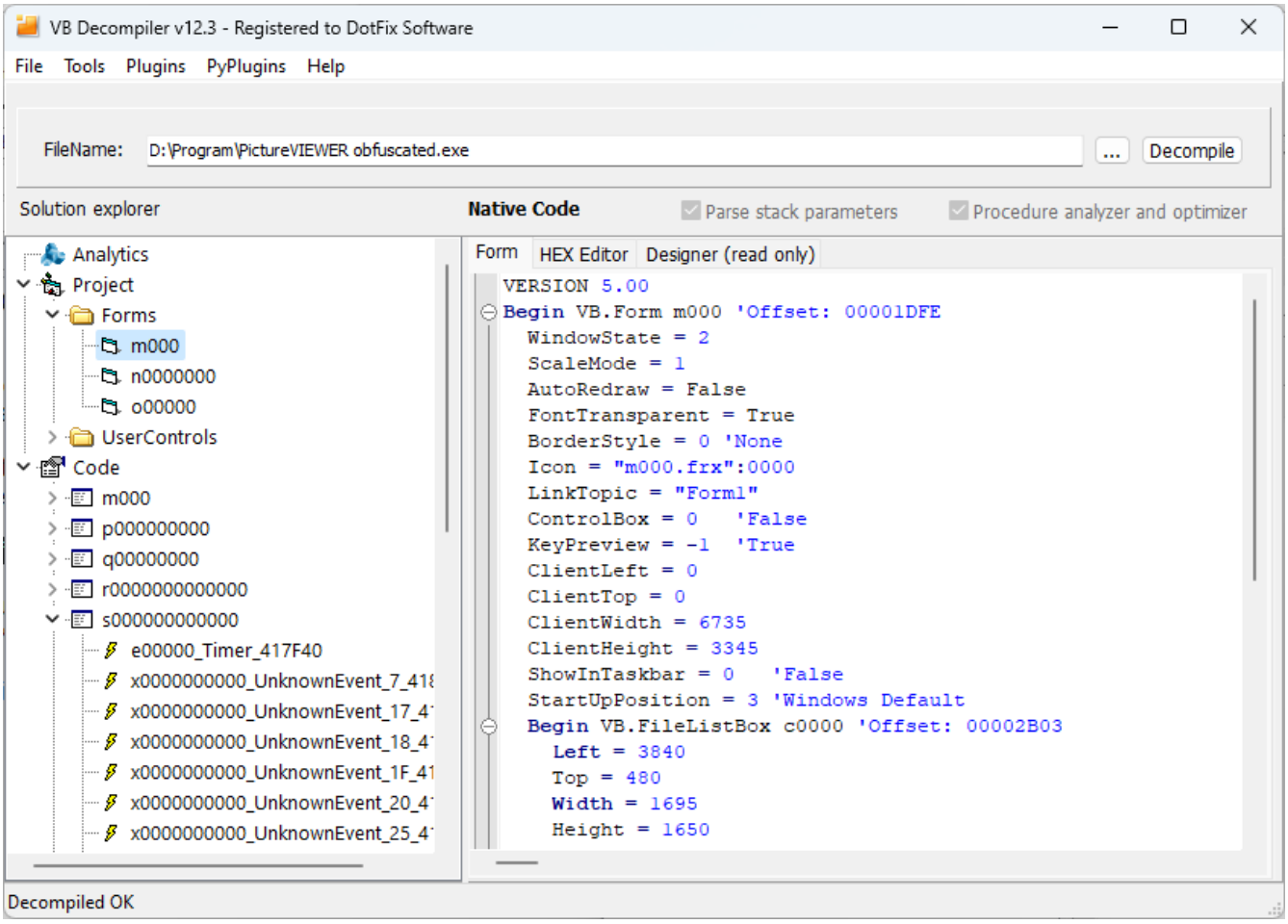
Patcher ile çalışmak son derece basitti. **Sanal Adres (Virtual Address)** alanında bir baytı düzenlemek için adresi girdikten sonra ya **Enter**'ye bastınız ya da **Al (Get)**'ye tıkladınız. Seçili adrese yakın konumda bulunan veri baytları Data alanında görüntülendi ve seçilen bayt değiştirilebildi; ardından **Ayarla (Set)**'ye tıklayarak girilen baytın değiştirilmesi gerekiyordu.

Eğer Program seçenekleri altında **Hızlı Geri Derleme (Fast decompilation)** seçeneği aktif ve **Önbellek Geri Derlenmiş Kodu (Cache decompiled code)** seçeneği devre dışıysa, tüm değişiklikler baytların değiştirildiği o işlevde yansıtılırdı. Bunun için, ağaçtaki adının üzerine çift tıklayarak aynı işlevi yeniden açmanız yeterliydi.

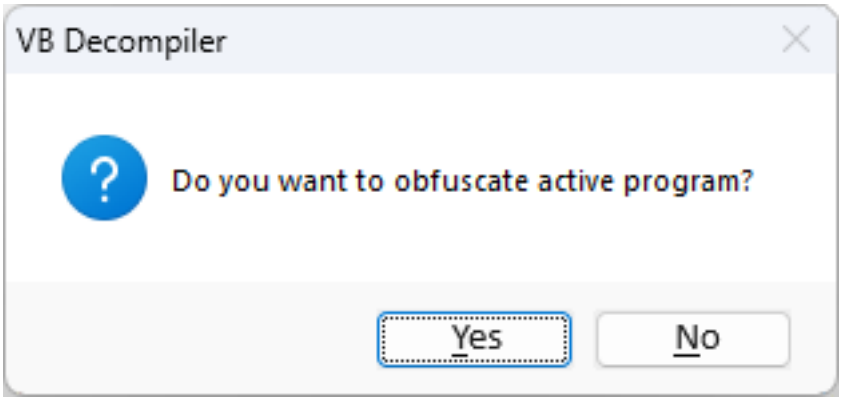
Önemli not: bu işlevsellik, yalnızca kendi program dosyalarınızı veya uygun yetkilere sahip olduğunuz dosyaları düzenlemek ve antivirüs laboratuvarlarında kötü amaçlı kod araştırmak için tasarlanmıştır. Başkasının kodunu değiştirmeden önce, böyle bir hakkın varlığına ilişkin lisans sözleşmesini gözden geçirdiğinizden emin olun.

7.6 Kod karartma

Visual Basic 5.0/6.0'de programlar geliştiriyorsanız, geri derlemeye karşı korumalarını artırabilirsiniz. Bunu sağlamak için, **Obfuscator**, programınızın ikili dosyasından tüm fonksiyon, form, modül ve nesne isimlerini çeşitli imzalar dahil olmak üzere kaldırabilecek şekilde geri derleyiciye gömülmüştür. Bu, sağlanan geri derleyici veya diğer çözümler kullanarak programı analiz etmeyi zorlaştırır. Ayrıca, çoğu üçüncü taraf ürün bu dosyayı tamamen geri derlemekten kaçınacaktır.



Bu korumayı uygulamak için, projenizi geri derleyicide açın ve **Projeyi Gizle (Obfuscate project)**'yi **Araçlar (Tools)** menüsünden seçin. Bu işlem çok fazla zaman almaz, bu nedenle projelerinizi yayınlamadan önce kod karartmayı öneririz.



Bu özellik, yazılım koruma için kapsamlı çözümümüz olan **DotFix NiceProtect**'ye de entegre edilmiştir ve www.niceprotect.com'dan indirilebilir.

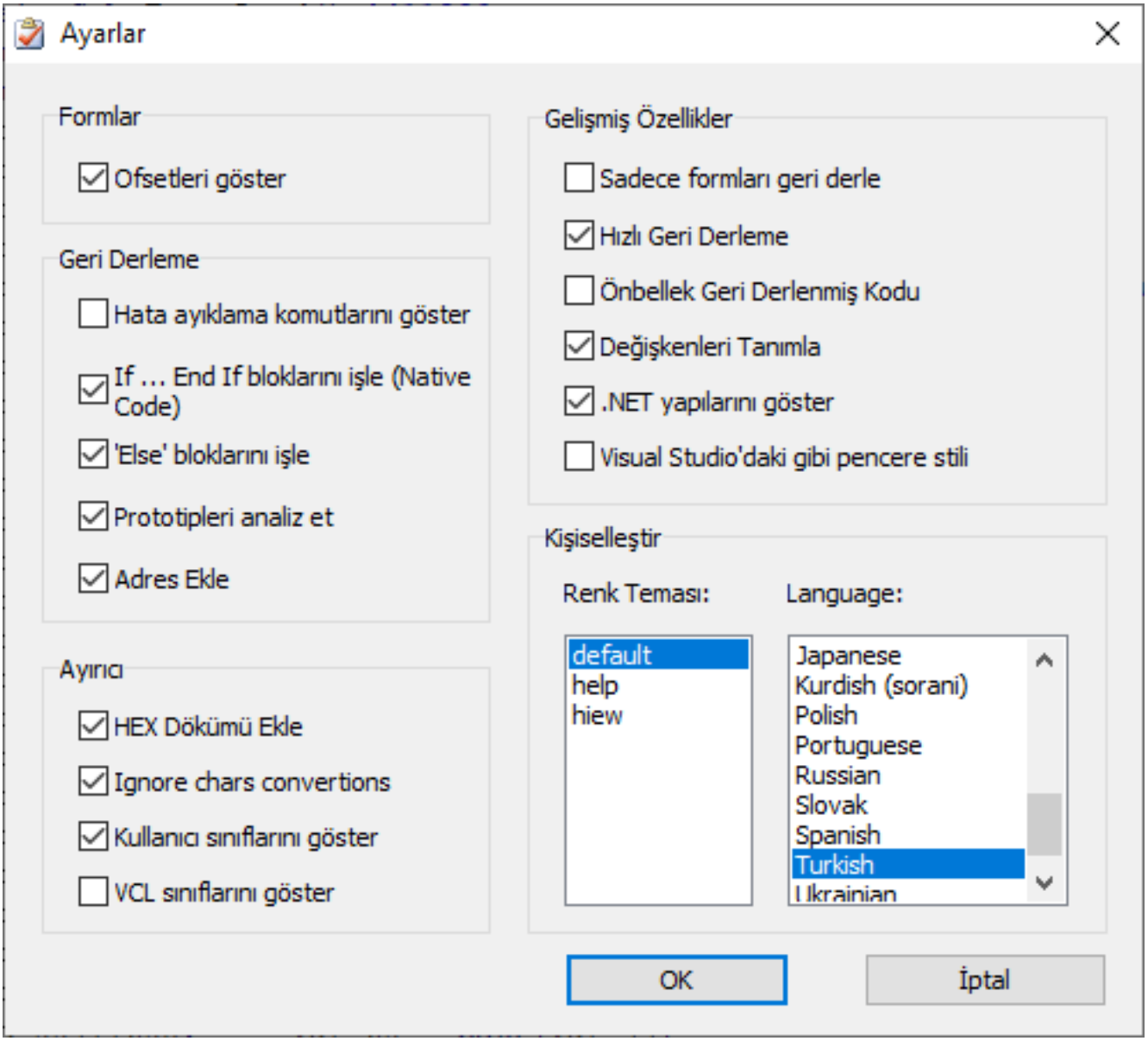
Kod karartma yalnızca Business License sahibi **VB Decompiler** kullanıcılarına sunulur.

Kod karartma kısıtlamaları

8 Program seçenekleri

8.1 Program seçenekleri

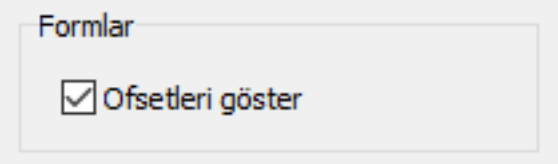
Program ayarlarını değiştirmek için **Araçlar (Tools)** menüsü > **Ayarlar (Options)** yolunu izleyin. Oluşan diyalogda aşağıdaki seçenekleri göreceksiniz:



Formlar
Geri derleme
Disasm. (Disassembler)
Renk düzeni Gelişmiş özellikler Dil seçimi

8.2 Formlar

Aşağıdaki geri derleme ayarlarını formlar için değiştirebilirsiniz:



Bu ayar bölümünde, form nesnelerinin yanına ofsetlerin gösterilmesini etkinleştirebilirsiniz. Bu, örneğin kaynak kodlara erişim olmadan başka bir dilde bir programı yerelleştirirken gerekli olabilir. **Ofsetleri göster (Show offsets)**, bu nesnelerin, isimlerinin ve özelliklerinin program içindeki konumlarını bulmanıza olanak sağlayacaktır.

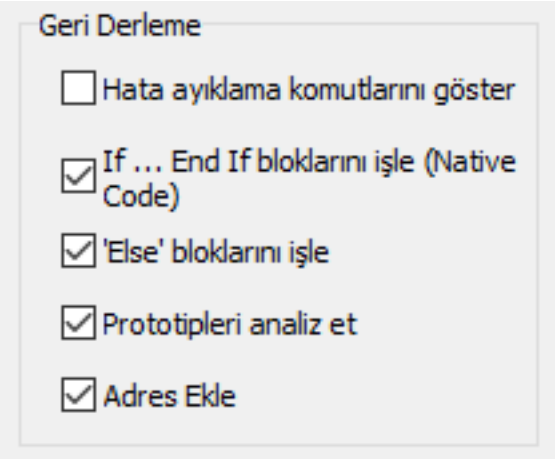
Daha sonra, bir HEX Düzenleyici kullanarak bu değerleri değiştirebilirsiniz. Ancak, bilginiz ikili dosyaların iç işleyişine aşina olmayan temel kullanıcı seviyesindeyse - bu seçeneği etkinleştirmeniz önerilmez. Dosyaya yapılan herhangi bir yanlış değişiklik onu işlevsiz hale getirecek ve gereksiz adresler sadece size engel olacaktır.

Görsel önizleme ve olay gezinmesi

Form seçenekleri **Tasarımcı (salt okunur) (Designer (read only))** görüntüleyicisinden ayrıdır. Klasik VB formları, Windows Forms, WPF/BAML desteği, olay-yöneticisi gezinmesi ve her arayüz teknolojisinin sınırlamaları için Görsel form yeniden inşası'na bakın.

8.3 Geri derleme

Geri derleme ayarlarını yapılandırmak için **Araçlar > Ayarlar (Tools > Options)** menüsünü açın. Ayarları etkileşimli olarak karşılaştırırken Önbellekleme olmadan hızlı geri derleme kullanın; aksi takdirde bir seçeneği değiştirdikten sonra dosyayı tekrar işleyin.



Hata ayıklama komutlarını göster (Show debug commands)

Klasik VB P-Kod'u için, yüksek seviyeli listeden genellikle atlanan derleyici ve hata ayıklayıcı işlemlerini içerir. Altındaki talimat dizisini yalnızca yeniden oluşturulan program mantığından ziyade incelemek istediğinizde bunu etkinleştirin.

If ... End If bloklarını işle (Native Code) (Process If ... End If blocks (Native Code))

Tanımlanan Native Code koşullu atlama işlemlerini `If...End If` blokları olarak yeniden oluşturur. Orijinal dallanma desenini incelemek veya yapısal olarak yanlış görünen bir yordamı araştırmak gerektiğinde bunu devre dışı bırakın.

'Else' bloklarını işle (Process 'Else' blocks)

Else dallarını iç içe koşullardan ve GoTo geçişlerinden ayırt etmek için ek sezgisel yöntemler kullanır. Native Code için bu, If bloklarının işlenmesini gerektirir. Normal okuma için bunu aktif tutun; yeniden oluşturulan bir dal belirsiz olduğunda devre dışı bırakıldığında karşılaştırın.

Prototipleri analiz et (Analyze prototypes)

Yordam parametreleri ve dönüş değerleri üzerinde sezgisel ön analiz yapar. Bu, hızlı geri derleme ile özellikle yararlıdır: standart modüllerdeki Private olarak bildirilmiş işlevler ve yordamlar açık imza meta verilerine sahip olmayabilir. Ön analiz ek zaman alır ancak çağrıyı yalnızca şu anda bilinen yığın içeriğine göre işlemeyi önler. Meta veri, dış türler veya kod eksik olduğunda sezgisel bir imza hala bir çıkarımdır.

Adres Ekle (Add Addresses)

Yeniden oluşturulan ifadelerin yanına sanal adresleri gösterir. Kodu makine kodu çözümlemesi, dosya baytları ve diğer araçlardan bulunanlarla ilişkilendirmek için bunu aktif tutun. Ayrıca **Disasm.'a kopyala (Copy to disassembler)**'nin yüksek seviyeli ifadeleri düşük seviyeli eşlemleriyle birleştirmesi için de gereklidir. Disasm. (Disassembler) bu ayardan bağımsız olarak her zaman adresleri gösterir.

Native Code yeniden oluşturmanın özellikleri

Native Code motoru desteklenen `Select Case` desenlerini, çok boyutlu dizilere erişimi, COM çağrılarını ve VB çalışma zamanı işlemlerini tanır. Hata yönetimi `On Error Resume Next`, `On Error GoTo` ve `On Error GoTo 0` olarak yeniden oluşturulabilir. Derleyici optimizasyonu ve eksik tür bilgisi hala sonucu sınırlayabilir; yüksek seviyeli çıktı eksik olduğunda makine kodu çözümlemesini inceleyin.

26.2 sürümü, Variant tiplemesini, çok boyutlu SafeArray okumalarını ve yazmalarını ve dizi elemanları için `VarPtr` kurtarmasını geliştirdi. Çıktı, analiz motoru tarafından tanınan çalışma zamanı yapısını yansıtan bir dizi referansını `SafeArray` olarak belirleyebilir. Dalların durumlarının ayrı ayrı izlenmesi ve geçici değişken analizi de karmaşık If/Else kodunun yeniden oluşturulmasını iyileştirir. 26.5 sürümü, Select Case ve VB5/VB6 çalışma zamanı fonksiyonu işlemini daha da iyileştirir.

COM ve ActiveX tür bilgisi

VB Decompiler öncelikle bilinen arayüzler için kendi yerleşik bilgi tabanını kullanır. Bilinmeyen bir dış COM nesnesi için, sistem kayıt defteri üzerinden kütüphane, sınıf ve arayüz tanımlayıcılarını çözebilir ve ilgili DLL/OCX tür kütüphanesini parse edebilir. Bu işlem, erken ve desteklenen geç bağlamalı çağrılar için döndürülen arayüz türlerini de içeren yöntem ve özellik imzaları sağlar.

Gerekli bir tür kütüphanesi yoksa veya kaydı çözümlenemiyorsa, bazı çağrılar tanımsız kalabilir, örneğin `UnkVCall` olarak. Eksik bağımlılıklar bu nedenle analiz makineleri arasında çıktıyı karşılaştırırken önemlidir. Genişletilmiş IID bilgi tabanı ofis otomasyonu, ADO/DAO, XML ve HTTP, betikleme, kabuk, mesajlaşma ve dizin hizmeti arayüzlerini içerir; her mümkün COM çağrısının çözüleceği anlamına gelmez.

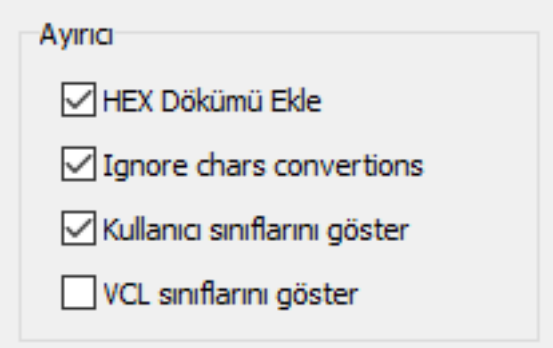
Bazı VB ikili dosyalarında yalnızca sınırlı miktarda UDT meta verisi korunur. Decompiler bir yapının boyutu ve string alan sayısı gibi bilgileri kurtarabilir ancak orijinal alan düzeninin tamamını kurtaramaz. Kısmi bir bildirimi tam kaynak seviyesi tanımı olarak yorumlamayın.

.NET yeniden oluşturma ve optimizasyon

Desteklenen IL, C# benzeri ifadeler ve kontrol akışı olarak yeniden oluşturulur. Ana pencerede bulunan **Prosedür Analizörü ve Optimize Edici (Procedure analyzer and optimizer)**, desteklenen döngü, koşullu, operatör ve geçici değişken desenlerini basitleştirebilir. Desteklenmeyen veya karartılmış bir parça kaldığında IL ve meta veriyi inceleyin. Bu işlem YZ yardım (AI Helper) işlevinden bağımsızdır.

8.4 Disassembler

Bu seçenek bloğu, makine kodunun çözümlenmesini denetler:



HEX Dökümü Ekle (Add HEX dump), analiz sonrası bir HEX Düzenleyici'sinde ikili dosyayı düzenlemeye gerek duyulduğunda gerekli olabilecek çözümlenen her makine komutundan önce makine kodlarını gösterir.

Karakter dönüşümü yok (Ignore chars convertions), İngilizce olmayan string verilerini sayısal Unicode'a dönüştürmeyi devre dışı bırakmak için kullanılır. Bu seçenek aktif tutulduğunda görüntülenen dizeler daha okunabilir hale gelir. Sayısal Unicode temsiline ihtiyaç duyduğunuzda bu seçeneği devre dışı bırakın.

Kullanıcı sınıflarını göster (Show user classes), **Delphi** / **Free Pascal** uygulamaları içinde kullanıcı sınıflarının ayrıştırılmasını içerir. Bu seçenek, **Visual Basic** ve **C#** uygulamaları için kullanılmaz.

VCL sınıflarını göster (Show VCL classes), **VCL** kütüphanesinin tüm iç sınıflarının geri derlenmesini içerir. Disassemblerün genellikle programları analiz ederken gereksiz olan binlerce hizmet sınıfını analiz etmesini engellemek için bu seçeneğin devre dışı bırakılması önerilir. Bu seçenek yalnızca **Delphi** uygulamaları için iç amaçlarla geri derleme yapmak üzere oluşturulmuş olup, diğer ürünlerle uyum sağlamak amacıyla tasarlanmıştır ancak **VB Decompiler**'de uygulanmamıştır.

Delphi ve **Free Pascal** uygulamalarının kısmi geri derlemesi şu anda yalnızca ayrı bir hizmet olarak (yazılım olarak değil) sunulmaktadır.

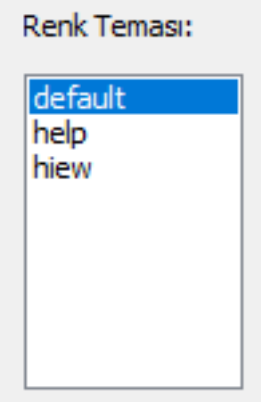
VB P-Kodu ve .NET IL

HEX Dökümü Ekle, düşük seviyeli P-Code listesini de destekler. Kurtarılan deyimleri P-Code veya yerel makine komutlarıyla aynı listede karşılaştırmak için Decompiler görünümünden **Disasm.'a kopyala (Copy to disassembler)** komutunu kullanın.

.NET derlemeleri için, Disasm. (Disassembler) uygun olan IL veya yerel x86/x64 yöntemlerini gösterebilir. Geri kazanılan yerel değişken imzaları diziler, genel türler, değer tipleri ve TypeSpec referanslarını içerebilir. Meta veri tabloları ve öbekleri kodla birlikte incelemeye ihtiyaç duyduğunuzda **.NET yapılarını göster (Show .NET structures)**'yi aktif hale getirin. Daha sonraki sürümler, FieldPtr tablosu dahil olmak üzere kodu karartılmış meta veri desteğini genişletir; bu, her hasarlı veya kodu karartılmış derlemenin başarıyla geri kazanılmasını garanti etmez.

8.5 Renk düzeni

Bu bölüm, geri derlenmiş kodu görüntüleme sırasında renk şemasını ve sözdizimi vurgulamayı tanımlar:



Önayarlar, VB Decompiler'ın yüklendiği dizindeki .ini dosyaları içindeki "colors" klasöründe saklanır. Dosya adları programın ayarlar bölümündeki önayar isimlerini belirler. Bu dosyaların basit bir formatı vardır ve kendiniz kolayca oluşturabilir veya mevcut olanları düzenleyebilirsiniz.

Yapılandırma formatı şu şekildedir:

```
[Colors]
Background = decimal color code (for example: 1973790)
Foreground = decimal color code (for example: 14671840)
KeyWords = decimal color code (for example: 13471849)
LocationsAndArguments = decimal color code (for example: 11849128)
Procedures = decimal color code (for example: 5676246)
; next properties is only available in version 12 or later
UnderCursorBackground = decimal color code (for example: 6986069)
UnderCursorForeground = decimal color code (for example: 1973790)
UseDarkMode = 0 or 1 (0 for Light Theme, 1 for Dark Theme)
Colors2 = 0 or 1 (using extended colors at Colors2 section)
```

```
[Font]
Name = font name (for example: Consolas)
Size = font size (for example: 11)
KeyWordsBold = 0 or 1 (font bold)
LocationsAndArgumentsBold = 0 or 1 (font bold)
ProceduresBold = 0 or 1 (font bold)
```

```
; this section is only available in version 12 or later
[Colors2]
TVFontName = Solution Explorer treeview font name or list of font names, separated by comma (for example:
    Segoe UI, Tahoma, Arial)
TVFontSize = size of the font (for example: 10)
FontName = Decompiler and Disassembler font name or list of font names, separated by comma (for example:
    Courier New, Consolas, Liberation Mono, monospace)
FontSize = size of the font (for example: 11)
Background = BackColor in hexadecimal for Decompiler and Disassembler tab (for example: 1E1E1E)
Foreground = ForeColor in hexadecimal for Decompiler and Disassembler tab (for example: DFDFE0)
; code highllighting
Key = color in hexadecimal (for example: CD9069)
Comment = color in hexadecimal (for example: CD9069)
Char = color in hexadecimal (for example: B4CDA8)
Float = color in hexadecimal (for example: B4CDA8)
Hex = color in hexadecimal (for example: B4CDA8)
Number = color in hexadecimal (for example: B4CDA8)
String = color in hexadecimal (for example: 569CD6)
Symbol = color in hexadecimal (for example: DFDFE0)
; HEX Editor font and colors
HexFontName = font name or list of font names, separated by comma (for example: Courier New, Consolas,
    Liberation Mono, monospace)
HexFontSize = size of the font (for example: 11)
HexBackground = color in hexadecimal (for example: 1E1E1E)
HexForeground = color in hexadecimal (for example: DFDFE0)
HexGrid = color in hexadecimal (for example: 1E1E1E)
HexActiveFieldBackground = color in hexadecimal (for example: 1E1E1E)
HexOffsetBackground = color in hexadecimal (for example: 1E1E1E)
HexCurrentOffset = color in hexadecimal (for example: B4CDA8)
HexOddColumn = color in hexadecimal (for example: B4CDA8)
HexEvenColumn = color in hexadecimal (for example: B4CDA8)
HexOffset = color in hexadecimal (for example: 569CD6)
```

Renk kodu, 32-bitlik gerçek renk formatında belirtilir. Colors bölümünde ondalık (eski VB Decompiler sürümleriyle geriye dönük uyumluluk için) ve Colors2'de hexadecimal.

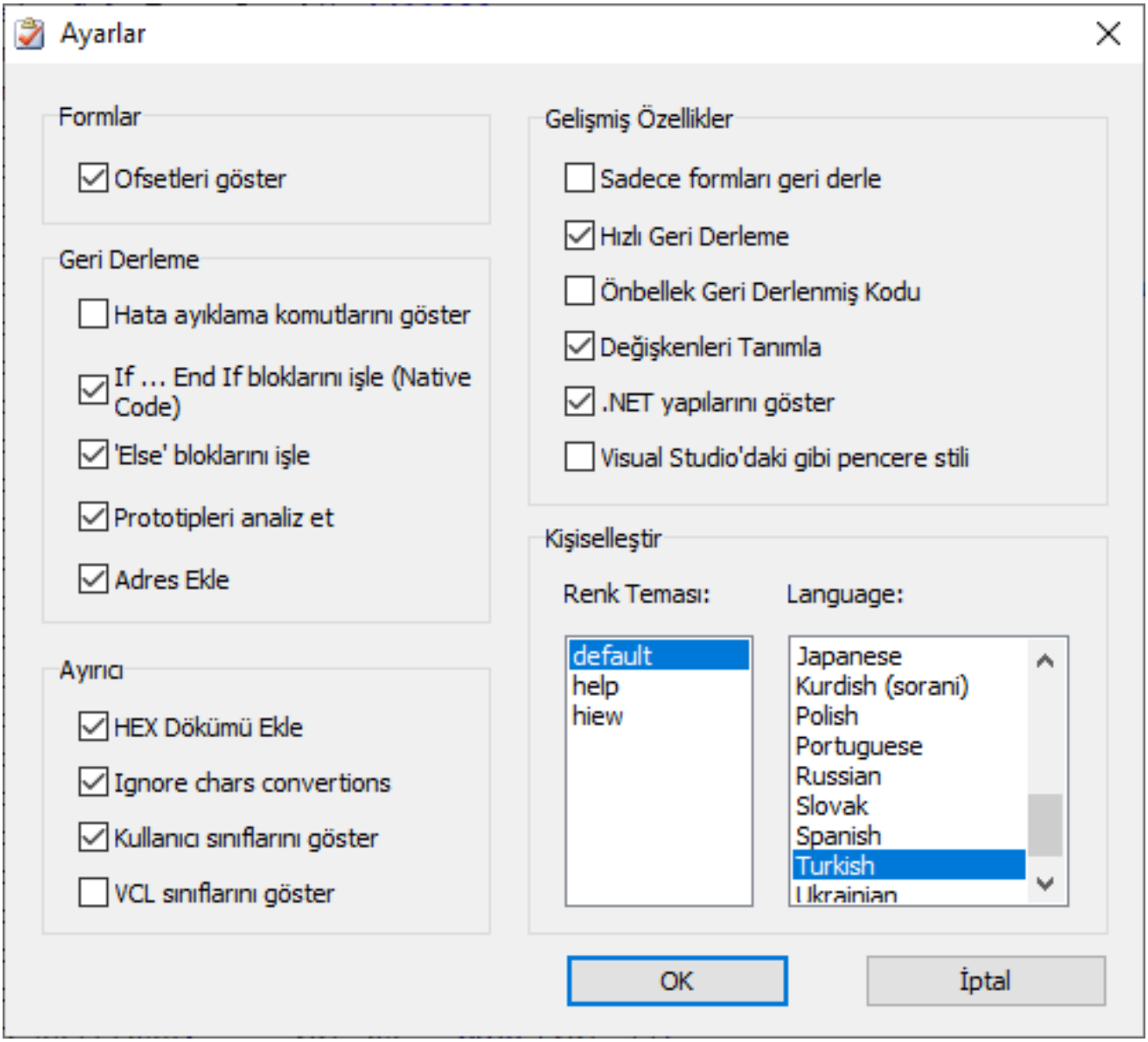
Kod Analizi Sırasında Temaları Değiştirme

Tema sistemi karanlık modu destekler ve VB Decompiler'ı yeniden başlatmadan, INI dosyasını düzenledikten sonra bile seçilen temayı uygulayabilir. Genişletilmiş yapılandırmadaki yazı tipi ayarları, eşdeğer yazı tiplerinin Windows ve uyumluluk ortamları arasında farklı olduğunda yardımcı olmak için virgülle ayrılmış yedekleme isimleri içerebilir.

İmleç altındaki renkler kimlik vurgulamayı kontrol eder. Yerel, üye ve küresel değişken vurgulama için Kodun Görüntülenmesi'na bakın.

8.6 Gelişmiş özellikler

Bu ayarlar projenin işlenişini ve geri kazanılan bilginin sunulma şeklini kontrol eder.



Sadece formları geri derle (Decompile only forms)

Sadece geri kazanılmış form ve UserControl tanımlamalarına ihtiyaç duyduğunuzda kod geri derleme işlemini devre dışı bırakır. Görsel görüntüleyici ve teknolojiye özgü yetenekleri görmek için Görsel form yeniden inşası sayfasına bakınız.

Hızlı Geri Derleme (Fast decompilation)

Aktif olduğunda, VB Decompiler proje yapısını önceden işler ve siz onları açtıkça bireysel yordamları geri derler. Bu, seçilen yordamları incelerken başlangıçtaki bekleme süresini azaltır. **Önbellek Geri Derlenmiş Kodu (Cache decompiled code)** devre dışı olduğunda, analiz ayarlarını değiştirebilir ve bir yordamı yeniden açarak yeni bir liste alabilirsiniz.

Prototipleri analiz et (Analyze prototypes) seçeneğini geri derleme ayarları bölümünde etkinleştirerek özel yordamların imzaları üzerinde sezgisel ön analiz yapın. Bu, ilk işlem süresini artırır ancak tam geri derleme gerektirmeden bağımsız değişkenlerin kurtarılmasını iyileştirebilir.

Hızlı Geri derleme, projenin tamamı için aranabilir yüksek seviyeli bir kod listesi oluşturmaz. Bunu devre dışı bırakın ve tam proje metin arama veya tam geri derlenmiş bir proje dışa aktarmadan önce dosyayı tekrar işleyin. String Referansları farklıdır: ön makine kodu çözümlemesi yoluyla toplanabilirler ve Hızlı Geri derleme modunda mevcuttur.

Tam geri derleme

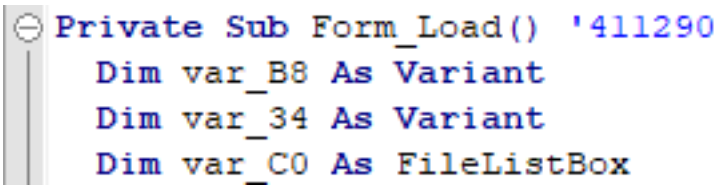
Hızlı Geri derleme devre dışı olduğunda, geri derleyici projenin yordamlarını önceden işler. Bu daha uzun sürer ancak tam oluşturulan liste metin arama ve dışa aktarma için kullanılabilir. Geri derleme ayarlarındaki değişiklikler dosyayı tekrar işlemeyi gerektirir.

Önbellek Geri Derlenmiş Kodu

Hızlı Geri derleme modunda zaten oluşturulan listeleri saklar. Bu, açılan bir yordamı tekrar tekrar geri derlemeyi önler ancak önbelleklenen sonuçlar sonraki analiz ayarı değişikliklerini yansıtmaz. **Yığın parametrelerini geri derle (Parse stack parameters)** veya **Prosedür Analizörü ve Optimize Edici (Procedure analyzer and optimizer)** ayarlarını etkileşimli olarak karşılaştırırken önbelleklemeyi devre dışı bırakın.

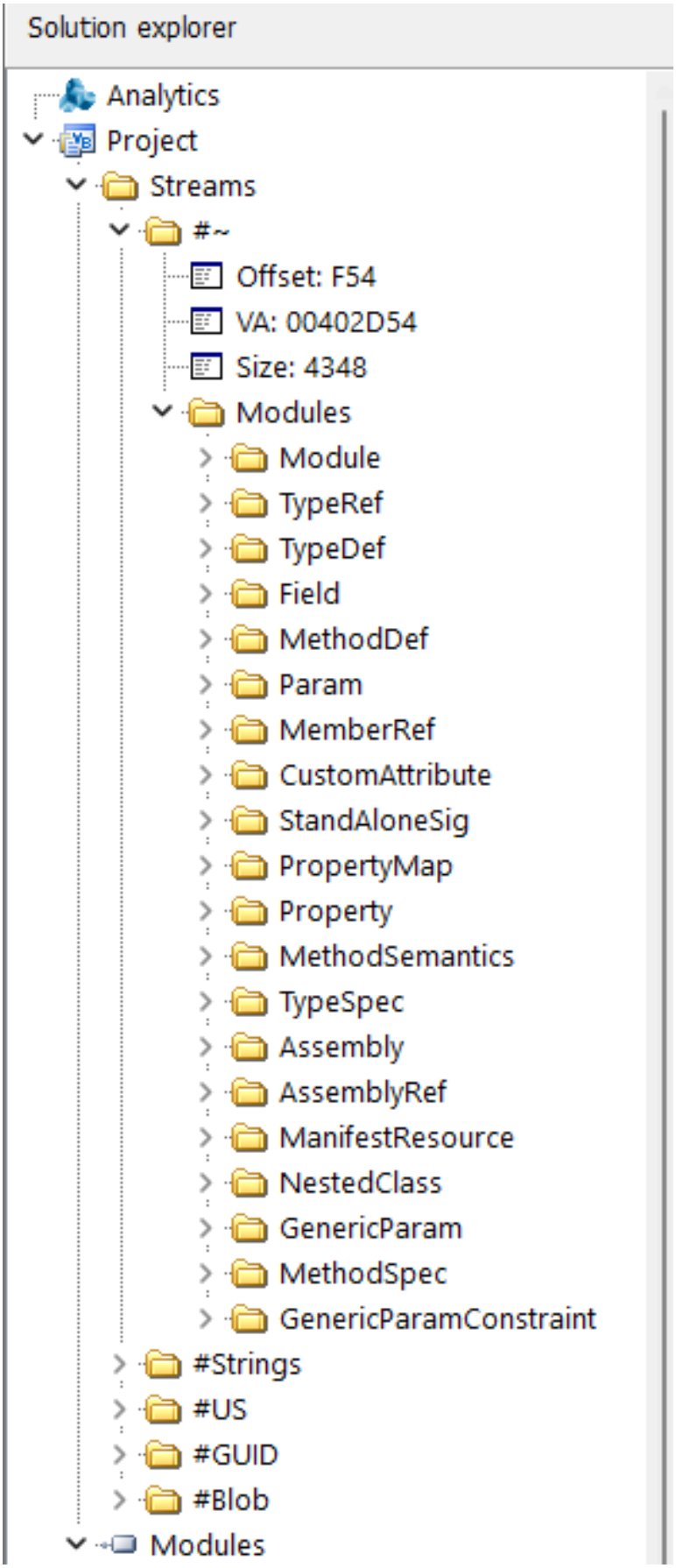
Değişkenleri Tanımla (Declare variables)

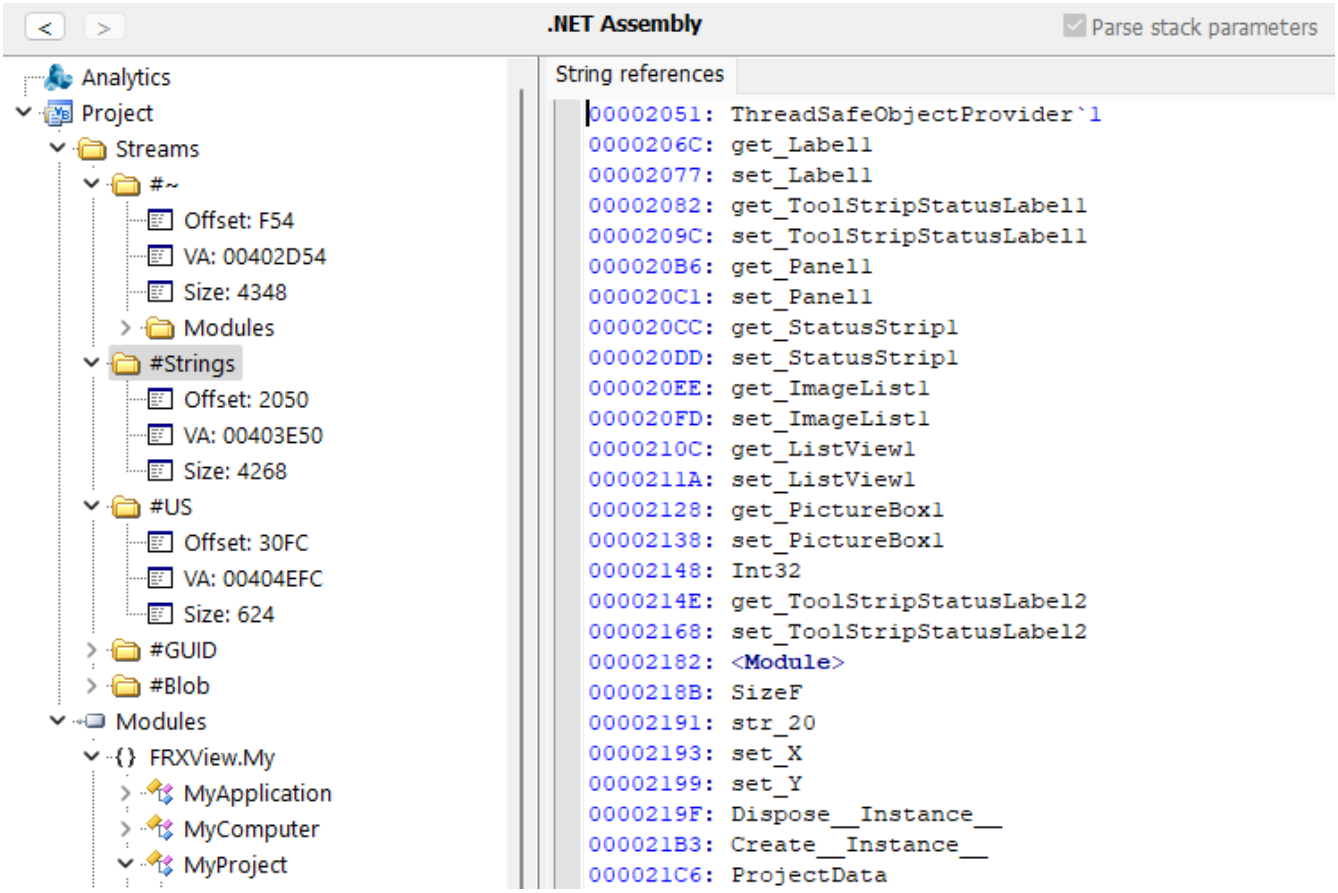
Her geri derlenmiş yordamın başında geri kazanılan değişkenler için bir bildirim bloğu ekler. Bu, kaynak kod kurtarma ve tip denetimi için kullanışlıdır. Kısa davranışsal bir inceleme için bildirimleri gizlemeyi tercih edebilirsiniz. Geri kazanılmış tipler, geri derleyicinin erişimine açık olan meta veri ve analize tabidir.



.NET yapılarını göster (Show .NET structures)

Çözüm Gezgini (Solution explorer)'da .NET meta veri tablolarını ve öbeklerini açığa çıkarır. Yapıları ve imzaları kodla birlikte incelemek için kullanın. Bir tabloyu seçmek içeriğini gösterir; #US öbeği kullanıcı dizelerini içerir. Meta veri gezinmesine ihtiyacınız olmadığında seçeneği devre dışı bırakın.


Solution explorer
Analytics
Project
Streams
#~
Offset: F54
VA: 00402D54
Size: 4348
Modules
Module
TypeRef
TypeDef
Field
MethodDef
Param
MemberRef
CustomAttribute
StandAloneSig
PropertyMap
Property
MethodSemantics
TypeSpec
Assembly
AssemblyRef
ManifestResource
NestedClass
GenericParam
MethodSpec
GenericParamConstraint
#Strings
#US
#GUID
#Blob
Modules



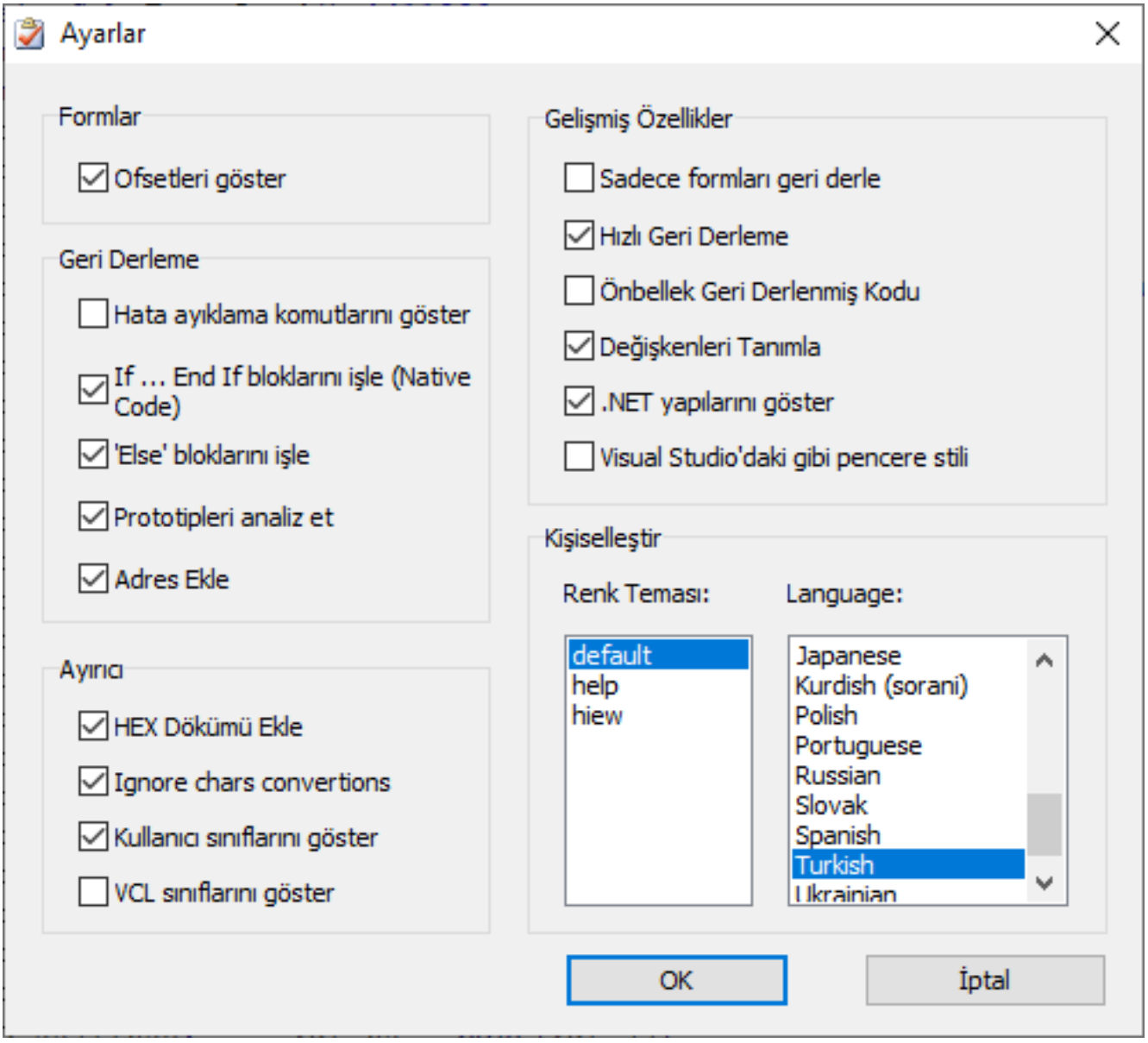
Visual Studio'daki gibi pencere stili (Window style as in Visual Studio)

Kodu sola, Çözüm Gezgini'ı sağa yerleştirir. Bu işlem, kurtarılan kodu değiştirmeden çalışma alanının düzenini değiştirir.

8.7 Dil seçimi

VB Decompiler, dünya genelinde kullanılan 20'den fazla dil için arayüz yerelleştirmelerini sağlar. Dokümantasyon da birden fazla dilde mevcuttur.

Programın tercih edilen dil arayüzünü, **Araçlar (Tools)** -> **Ayarlar (Options)** menüsünün altında bulunan **Kişiselleştir (Personalize)** ayarları bölümünde seçebilirsiniz. Dili seçtikten sonra, dil ayarlarının programın tamamı boyunca doğru bir şekilde uygulanması için **VB Decompiler**'yi yeniden başlatmanız gerekir. Programı yeniden başlatmak istenmiyorsa, **VB Decompiler**'i kullanmaya devam edebilirsiniz; ancak bazı metinler hala önceki dilde görüntülenebilir.



Yardım dili, bilgisayarınızın sistem diline göre otomatik olarak seçilir. Kılavuzu başka bir dilde okumak için, **VB Decompiler**'nin yüklü olduğu klasörden ilgili yardım dosyasını açın. Yerelleştirilmiş kılavuzların ana formatı PDF'dir; CHM yardım dosyaları bazı diller için hala mevcuttur.

İngilizce kılavuz, tüm çevirilerin kaynağıdır. Yerelleştirilmiş bir kılavuzda hata veya tutarsızlık fark ederseniz, lütfen teknik destek ekibine bildirin.

Kılavuza erişim

Yerel bir yardım dosyası veya gerekli PDF görüntüleyici mevcut değilse, VB Decompiler çevrimiçi dokümantasyonu açabilir. Çevrimdışı bir araştırma ortamı için, gerekli yerel yardım dosyalarını önceden hazır bulundurun.

9 Eklentiler

Eklentiler, yerel DLL'ler aracılığıyla VB Decompiler'i genişletir. Yüklenen eklentiler **Eklentiler (Plugins)** menüsünden erişilebilir. Geri kazanılmış proje bilgilerine ve koda erişim sağlayan geri çağrı API'si, bir eklentinin desteklenen iç yapıları incelemesine veya değiştirmesine olanak tanır.

VB/DE Decompiler MCP Server 2.1.0

Bir MCP istemcisini kod çözücüde açık olan projeye bağlayın. İncelenen programı çalıştırmadan modülleri, formları, yordamları ve elde edilen kodu okuyabilirsiniz.

x86 ana uygulamalar için MCP Server 2.1.0'i indirin · Delphi kaynak kodu

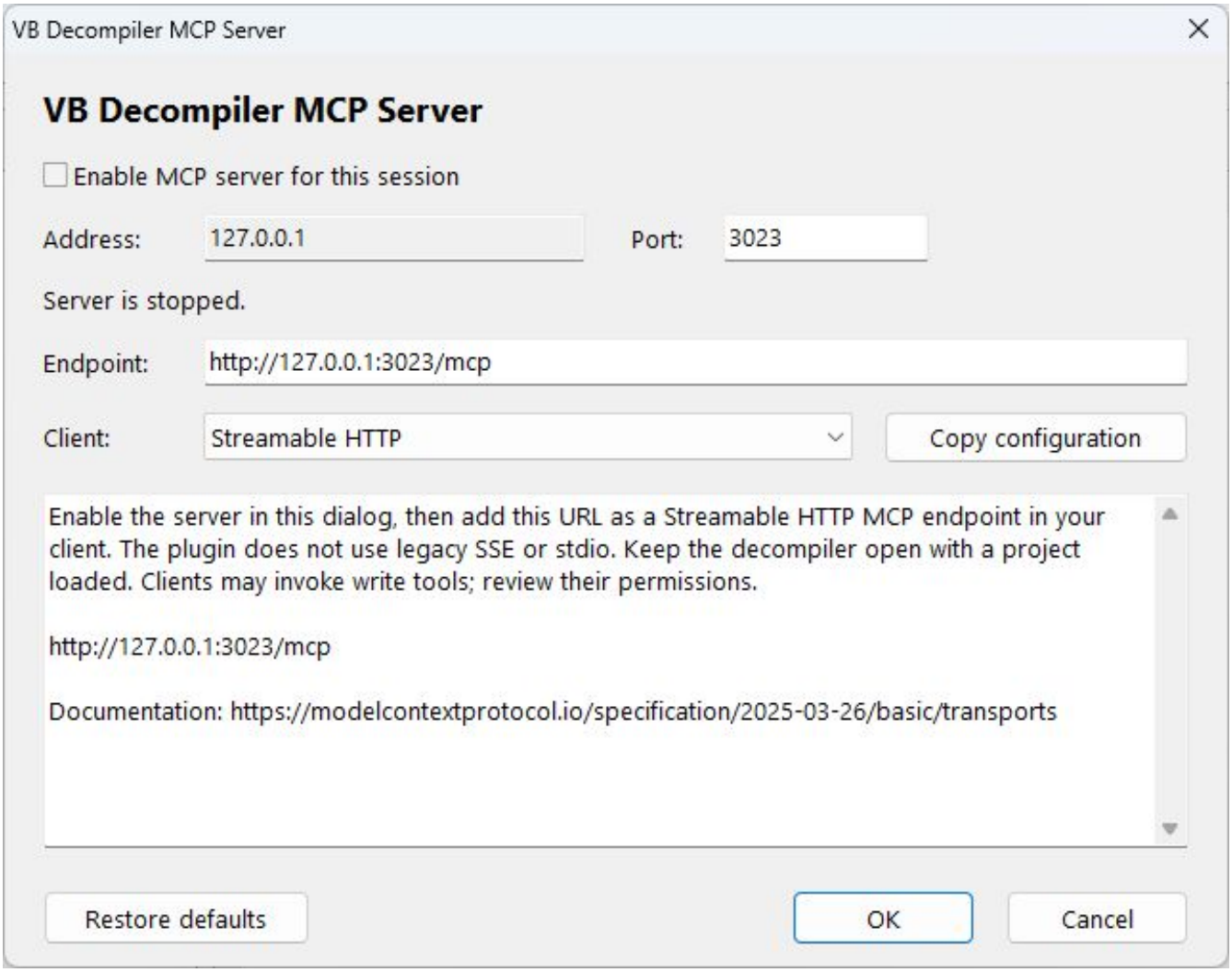
`vb_get_form`, form dizini `form_index`, form adı `form_name` ve özellik metni `form_text` alanlarını döndürür. Ad kullanılamıyorsa `form_name` boş bir dizedir. Birimleri ve sınıfları `vb_list_modules` ve `query` ile, işlev gövdelerini `vb_search_module_code` ile arayın. MCP komut başvurusu ve örnekler

Gereksinimler

Arşiv, 32 bit Windows kod çözücüsü için x86 eklentisini içerir. Araçların kullanılabilirliği sürüme, ürün edisyonuna ve açık projeye bağlıdır. Eski sürümlerde bazı işlemler yalnızca önbellekteki kodu okuyabilir.

Kurulum ve başlatma

1. Kod çözücüyü kapatın. Arşivi açın ve `vbmcp.dll` dosyasını kurulum dizinindeki `plugins` klasörüne kopyalayın. Eski bir DLL varsa değiştirin.
2. Kod çözücüyü başlatın ve bir proje açın. **Plugins** → **VB/DE Decompiler MCP Server** seçeneğini açın.
3. **Enable MCP server for this session** seçeneğini işaretleyin, **OK** düğmesine basın ve gösterilen adresi kullanın.



VB Decompiler uygulamasındaki MCP Sunucusu ayarları. Eklenti, kendisini yükleyen kod çözücüyü algılar ve pencere başlığında ürün adını gösterir.

MCP istemcisini bağlama

Streamable HTTP kullanın. Varsayılan adres `http://127.0.0.1:3023/mcp` şeklindedir. İki ürün aynı anda çalışıyorsa ayrı bağlantı noktaları seçin.

Codex

```
codex mcp add vbmcp --url http://127.0.0.1:3023/mcp
```

Claude Code

```
claude mcp add --transport http vbmcp http://127.0.0.1:3023/mcp
```

VS Code

VS Code'da **MCP: Add Server** komutunu çalıştırıp HTTP'yi seçebilirsiniz.

```
{
  "servers": {
    "vbmcp": {
      "type": "http",
      "url": "http://127.0.0.1:3023/mcp"
    }
  }
}
```

İstemci belgeleri · Codex · Claude Code · VS Code

Bağlantıyı sınama

Formlar içeren bir örnek açın ve `limit` değeri `5` olan `vb_list_forms` aracını çağırın. Bu sürümdeki araç adları `vb_` ile başlar.

Yerel erişim ve proje değişiklikleri

Sunucu yalnızca `127.0.0.1` adresini dinler ve kimlik doğrulaması yapmaz. Yerel istemciler proje verilerini okuyabilir veya kod listelerini, adları ve işaretleri değiştirebilir. İstemcide yazma onaylarını açık tutun. Sunucuyu durdurmak için işareti kaldırıp **OK** düğmesine basın. Her başlatmada sunucu kapalıdır.

Eklenti yerel olarak çalışır. MCP istemcisi kendi ayarlarına göre sonuçları model sağlayıcısına gönderebilir.

Delphi ile Sergey Chubchenko (DotFix Software) tarafından geliştirildi. Başlangıç noktası özgün MCP eklentisi oldu. MIT lisansıyla dağıtılır.

FRX Image Viewer

FRX Image Viewer 1.2.0, VB Decompiler veya DE Decompiler tarafından kaydedilen görüntüleri önizler ve dışa aktarır. VB5/6 ve .NET form grafikleri için VB Decompiler, ikili Delphi DFM grafikleri için DE Decompiler kullanın. BMP, PNG, JPEG, GIF, ICO ve CUR desteklenir.

Kurulum ve açma

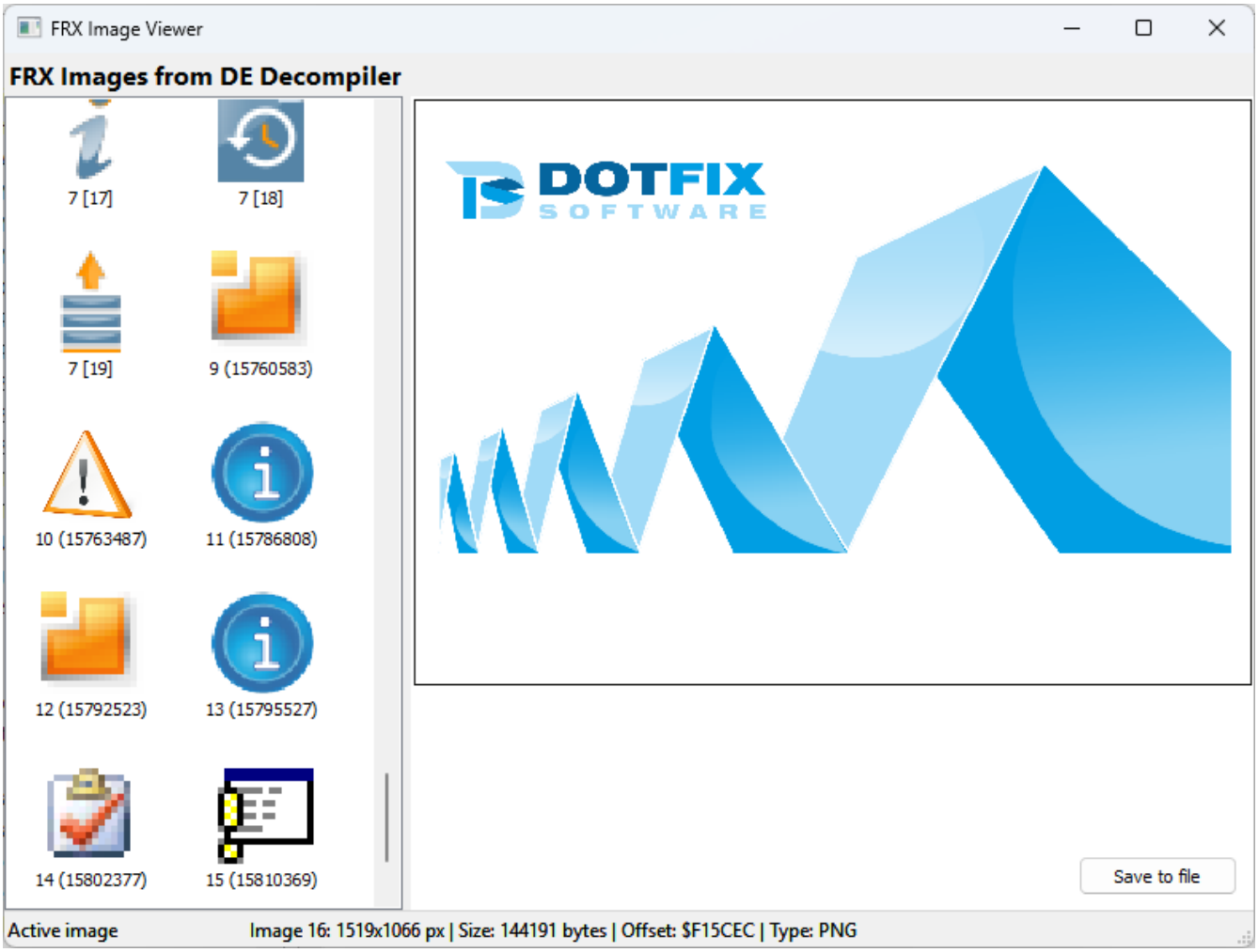
1. `FRXViewer.dll` dosyasını ana uygulamanın kurulumundaki `plugins` alt klasörüne kopyalayın ve decompiler uygulamasını yeniden başlatın. DLL mimarisi, incelenen dosyanın mimarisinden bağımsız olarak ana uygulamayla eşleşmelidir.
2. EXE veya DLL dosyasını açıp formlarını çözümleyin; böylece ana uygulama kullanılabilir görüntüleri kaydeder.
3. **Eklentiler** > **FRX Image Viewer** öğesini seçin, ardından bir küçük resme tıklayın.

1.2.0 indirme paketi, x86 ana uygulama için bir DLL içerir. x64 ana uygulama için uygun bir derleme gerekir. Eklenti Windows 10/11 gerektirir; derleme yönergelerini kaynak kod deposu üzerinden edinebilirsiniz.

Önizleme ve dışa aktarma

Galeri ve önizleme, çoklu görüntü içeren ICO/CUR dosyaları dahil kullanılabilir görüntüleri gösterir. Durum alanında öğe, boyutlar, veri boyutu veya kaynak bitmap konumu ve biçim gösterilir. Görüntüleyici ana uygulamanın temasını izler ve koyu modu destekler.

Seçili öğeyi dışa aktarmak için **Save to file** düğmesini kullanın. Bağımsız görüntülerin özgün baytları ve biçimi korunur. Ayrı ImageList hücreleri maske ve alfa saydamlığıyla ICO olarak dışa aktarılır; bu ICO dosyaları kaynak bitmaplerden üretilir.



DE Decompiler içinde Delphi DFM grafikleri ve ayrı ImageList simgeleri gösteren FRX Image Viewer.

Yerel Delphi ImageList desteği ve uyumluluk

Uyumlu bir ana uygulamada tanınan yerel Delphi ImageList verileri ayrı simgeler olarak görünür. Koleksiyon öğelerinde ayrı kodlanan resimler normal bağımsız görüntüler olarak kalır. Desteklenmeyen akışlarda özgün görüntü girdileri varsa tüm görüntüyü gösterme davranışı korunur.

VB Decompiler 26.6 ve DE Decompiler 26.5 tarafından desteklenen ortak SDK 26.6 sürümü, isteğe bağlı `GetFrxImageListInfo` (63) ve `GetProductName` (64) komutlarını ekler. Eski ana uygulamalarda özgün görüntü API'si ve tüm görüntü davranışı korunur. Özellik algılama ve konum kuralları için SDK başvuru belgesine bakın.

Önizleme ve dışa aktarma, incelenen dosyayı çalıştırmaz veya bileşenlerini yüklemez. Hareketli GIF dosyalarının ilk karesi gösterilir. Kullanılabilir kaynaklar, ana uygulamanın başarıyla çözümleyip kaydettiği görüntülere bağlıdır.

Eklenti Geliştirme

SDK, Visual C++, Rust, Delphi ve Free Pascal için örnekler içerir. Dışa aktarılmış giriş noktaları, geri çağrı imzası, komut tanımlayıcıları, veri temsilleri ve örnek projelere bağlantılar için VB Decompiler Eklenti API referansı'na bakın.

Bir DLL eklentisi adını ve yazar bilgilerini sağlar ve `VBDecompilerPluginLoad` üzerinden ana geri çağrıyı alır. Geri çağrı, desteklenen proje verilerini istemek veya güncellemek için kullanılır. Bu arayüzü uygularken SDK'nın çağrı kuralları, işaretçiler, dize kodlaması ve arabellek gereksinimlerine uyun; yönetilen bir dil kullanıyorsanız birlikte çalışabilirlik katmanının yerel arayüzle uyumlu olmasını sağlayın.

VB Decompiler, başlangıçta mevcut DLL eklentilerini başlatır. Eklentileri hedef konak mimarisi ve Windows ortamı için derleyin ve oluşan DLL'yi o ortamla doğrulayın. SDK, hem aktif-görünüm koduna hem de proje-seviyesinde kod

erişimini sunar; tüm geri derlenmiş yordamları gerektiren işlemler için tam geri derleme, yalnızca talep üzerine bir liste yerine gerekir.

10 Formların görsel olarak yeniden oluşturulması

Tasarımcı (salt okunur) (Designer (read only)) sekmesi, analiz edilen ikili dosyadan desteklenen arayüzlerin görsel bir yeniden inşasını gösterir. Geri kazanılan her özellik veya başlatma ifadesini manuel olarak yorumlamadan, arayüz öğelerini uygulama davranışıyla ilişkilendirmeye yardımcı olur.

Geri kazanılmış bir formu açma

1. Uygulamayı açın ve VB Decompiler'ın proje yapısını geri kazanmasını sağlayın.
2. Çözüm Gezgini (Solution explorer)'nde geri kazanılmış bir formu seçin ve **Tasarımcı (salt okunur)**'yi açın.
3. Mevcut özelliklerini ve geri kazanılan olay bilgilerini incelemek için bir kontrol seçin. İlgili işleyici koduna gitmek için olayları kullanın.

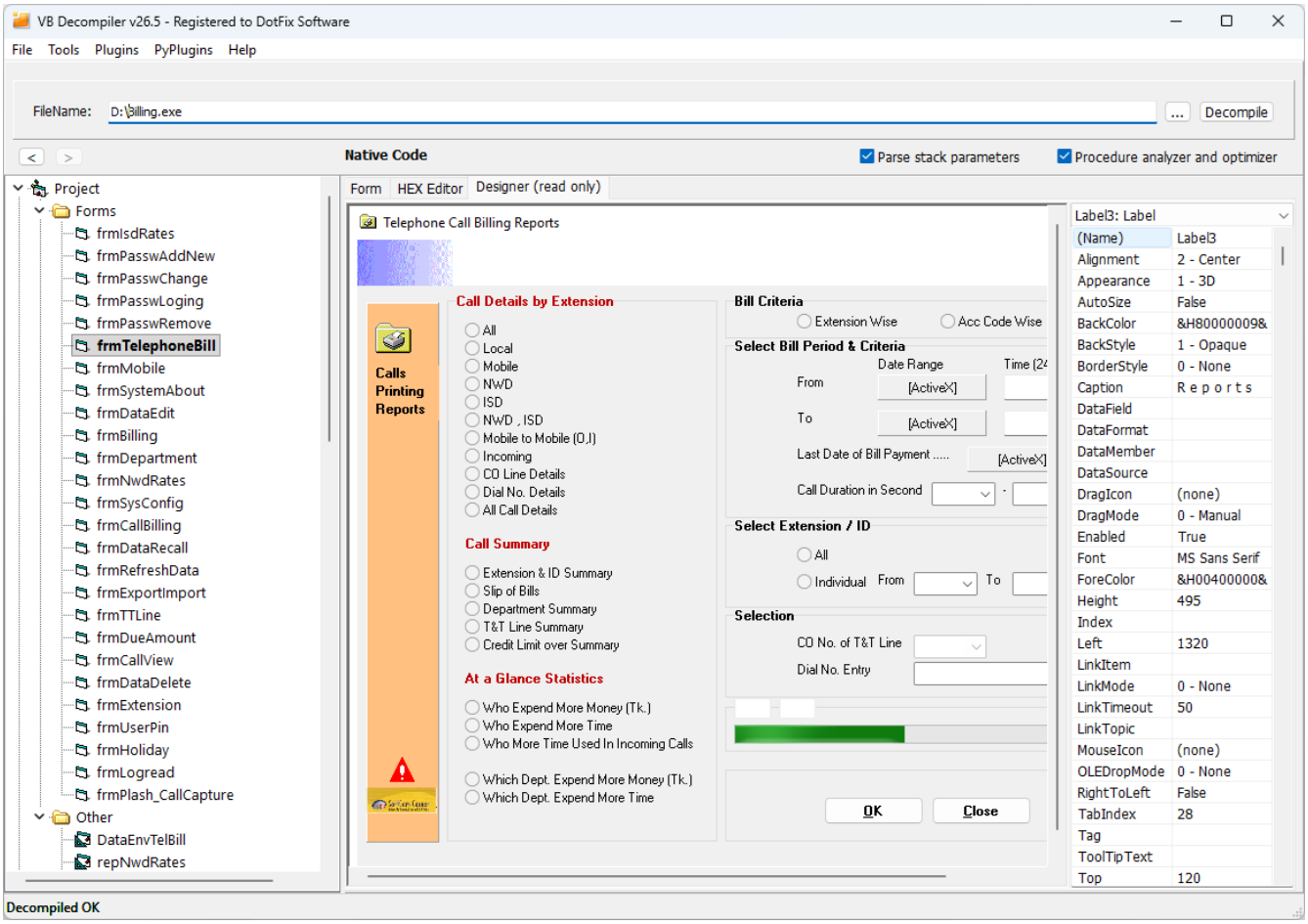
Bu önizleme, saklanmış bilgilerden yeniden inşa edilir. Orijinal arayüzün çalışan bir örneği değildir. Desteklenmeyen kontroller, çalışma zamanında oluşturulan içerikler, eksik kaynaklar ve özel yerleşim mantığı, eksik veya yaklaşık sonuçlara yol açabilir.

Visual Basic 5.0/6.0

Görüntüleyici, desteklenen kontrol özellikleri, yazı tipleri, renkler ve görseller dahil olmak üzere formları ve UserControls'u yeniden inşa eder. Listeler, açılır kutular ve diğer kontroller tarafından kullanılan ikili form verilerini okur. Görseller arasında simgeler, imleçler, bitmap'ler, JPEG, PNG ve GIF verileri bulunabilir.

Görsel menü yeniden inşası hem P-Code hem de Native Code uygulamalarını destekler. Desteklenen kontroller `SSTab`, `MSComctlLib.StatusBar`, `MSComctlLib.ListView`, `MSComctlLib.ProgressBar` ve `MSComctlLib.TreeView` içerir.

Klasik VB kontrolünün işleyicisini incelemek için, kontrolü seçin ve özellik listesindeki **_Event** alanına çift tıklayın. Oluşturulan listeden gerekli olayı seçerek işleyiciyi açın.

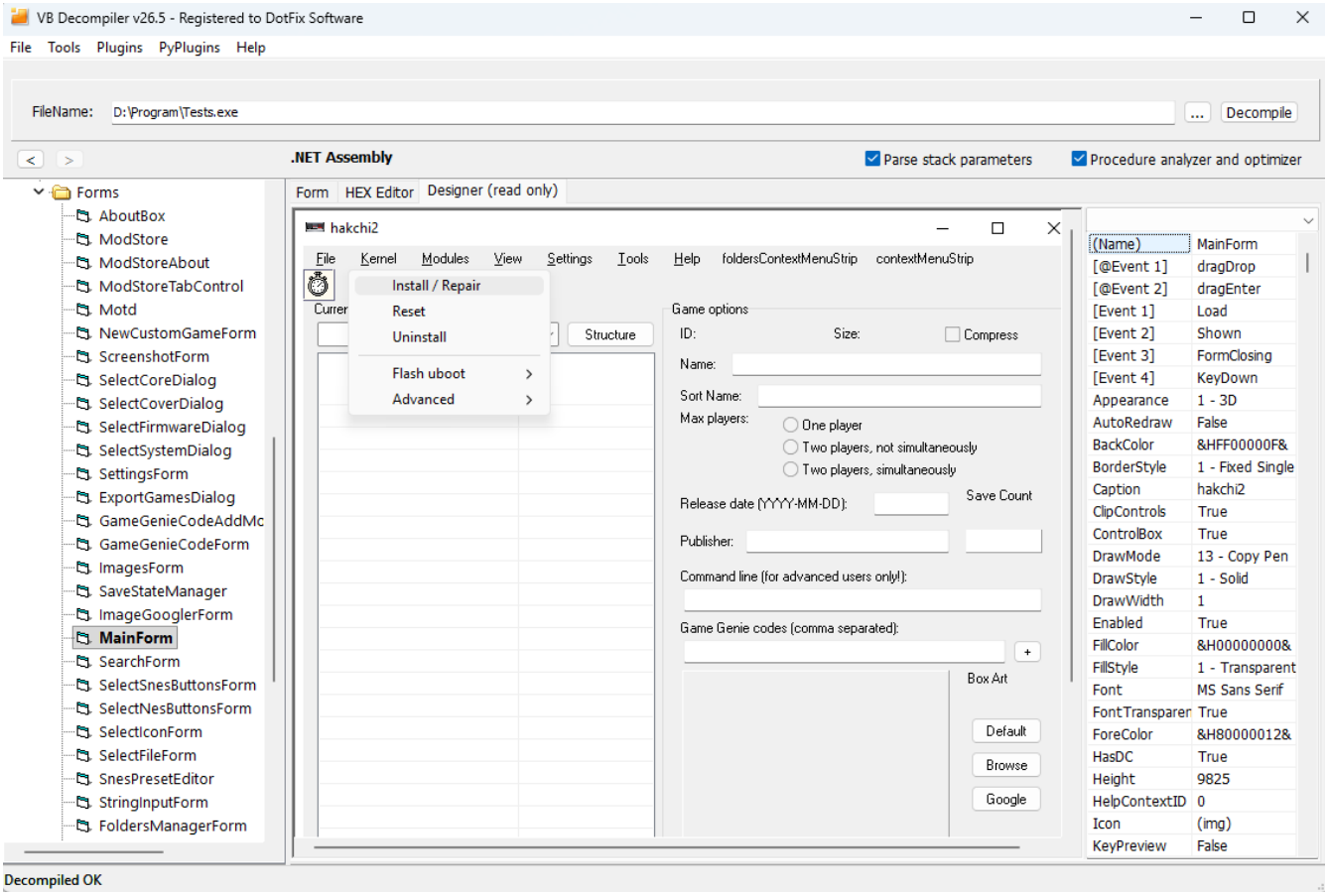


.NET Windows Forms

VB Decompiler, `InitializeComponent`'i statik olarak analiz eder ve desteklenen Windows Forms kontrollerini ve yerleşimlerini yeniden inşa eder. Ayrıştırıcı ve görüntüleyici, .NET runtime'ı veya hedef uygulamanın çalıştırılmasını gerektirmez. Geri kazanılmış desteklenen görseller ayrıca kontrollerde ve form arkaplanlarında da gösterilebilir.

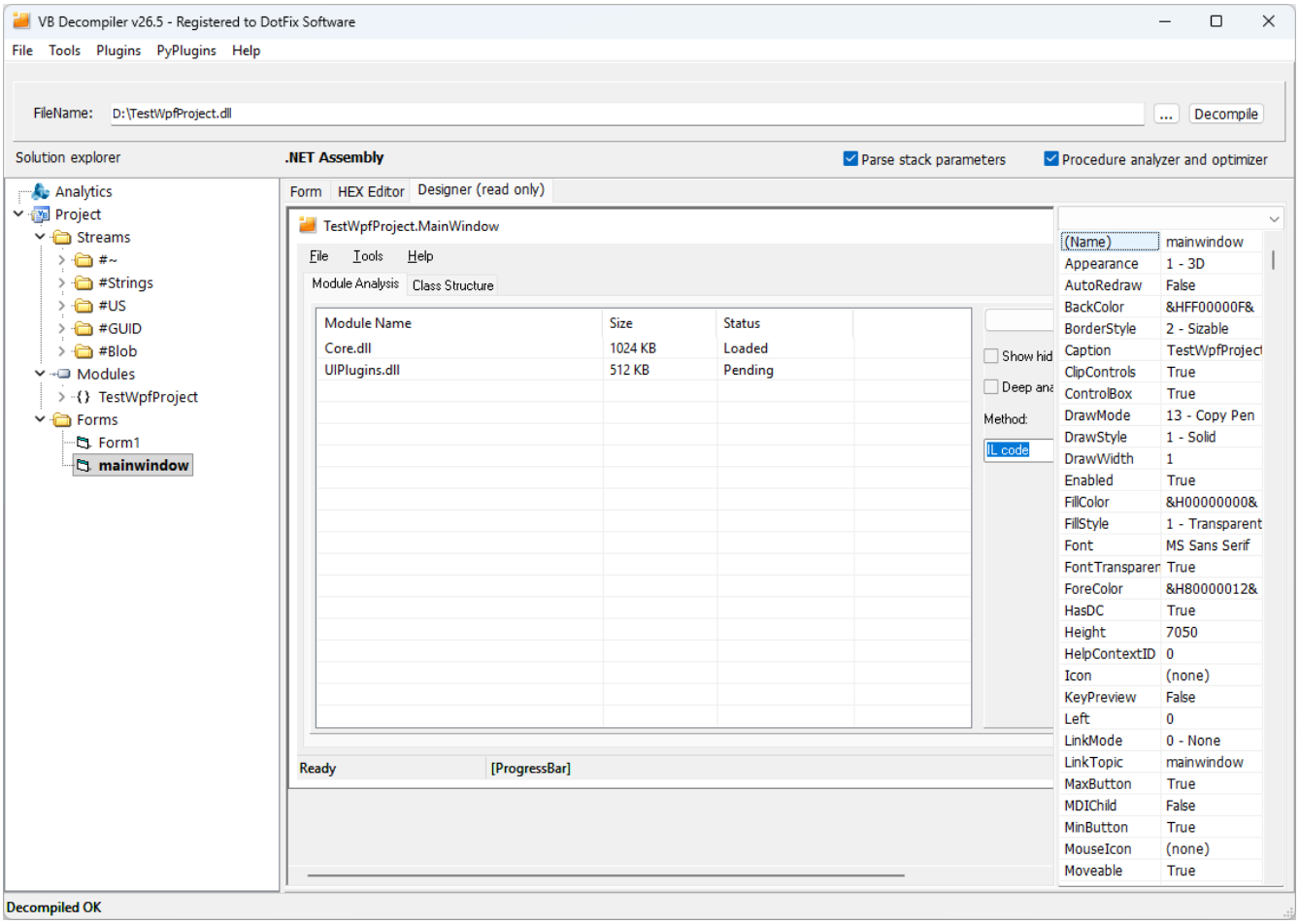
Kontrolü seçerek geri kazanılan olaylarını inceleyin, ardından bir olaya çift tıklayarak ilgili işleyiciyi açın. Görüntüleyici menü ve bağlam menüsü hiyerarşilerini yeniden inşa eder; geri kazanılmış bir menü öğesine tıklamak ilgili Click olayını açar. Özel Windows Forms olayları @ öneki ile gösterilir.

Görüntüleyici, kontrollerin Z-sırasını, devralınan ön ve arka plan renklerini, `Color.FromArgb` değerlerini, sayısal `FontStyle` bayraklarını, Boolean kaynak özelliklerini ve `MinimumSize` ve `ClientSize` özelliklerini işler.



WPF pencereleri, sayfaları ve kullanıcı kontrolleri

Görüntüleyici, **BAML** olarak saklanmış derlenmiş XAML kaynaklarından desteklenen WPF pencerelerini, sayfalarını ve kullanıcı kontrollerini yeniden inşa eder.



Desteklenen yerleşim konteynerleri arasında sabit, Auto ve oransal (*) satır ve sütun boyutları olan **Grid**, **StackPanel**, **DockPanel** ve **Canvas** yer alır; bunlar iç içe geçmiş yerleşimleri de içerir. Yeniden oluşturma sırasında desteklenen boyutlar, kenar boşlukları, dolgu alanı, hizalama, yönlendirme, kenara yaslama ve Grid satır/sütun uzanımları dikkate alınır.

Yaygın giriş kontrollerinin ötesinde, görüntüleyici sekmeleri, grup kutularını, genişleticileri, menü hiyerarşilerini ve ayırıcıları, durum çubuklarını, ListView sütunlarını ve desteklenen statik verileri, statik TreeView düğümlerini ve statik açılır kutu öğelerini yeniden inşa edebilir. Ribbon arayüzleri, gezinme panelleri ve kenara yaslanmış paneller basitleştirilmiş temsiller içerebilir.

Kaynak verisi mevcut olduğunda, görüntüleyici adları, görünen metinleri, ipuçları, yazı tipleri, arka plan renklerini, etkin ve görünür durumları ve gömülü görselleri (desteklenen PNG şeffaflığı ve arka plan opaklığı dahil) geri kazanır. Desteklenmeyen özel kontroller, orijinal WPF tipini tanımlayan yer tutucular olarak görünebilir.

WPF görüntüleyicisi, kurtarılan denetim olaylarını ilgili işleyicilerle ilişkilendirir. Karşılık gelen geri derlenmiş kodu açmak için görsel görüntüleyicideki olay bilgilerini kullanın.

Yeniden oluşturma sınırlamaları: görsel önizleme yaklaşık bir sonuçtur. Şablonlar, stiller, bağlamalar, özel kontroller ve dinamik olarak oluşturulan içerik kısmen temsil edilebilir.

Form verilerini inceleme ve çıkarma

Ofsetleri göster (Show offsets) ayarı ve bağlı ikili denetimi için Formlar'a bakın. Desteklenen geri kazanılmış görsel varlıkları incelemek ve dışa aktarmak için FRX Görüntüleyici Eklentisi'nı kullanın.

11 İzleme

Tracing features, Visual Basic 5.0/6.0 Native Code ve P-Code için adım adım öykünme sağlar. Bu işlev için izleme özelliklerini içeren bir lisans gerekir. İzleyici, yordamın öykünücüdeki iç gösterimi üzerinde çalışır; hedef yordamı iş istasyonunun CPU'sunda yürütmez.

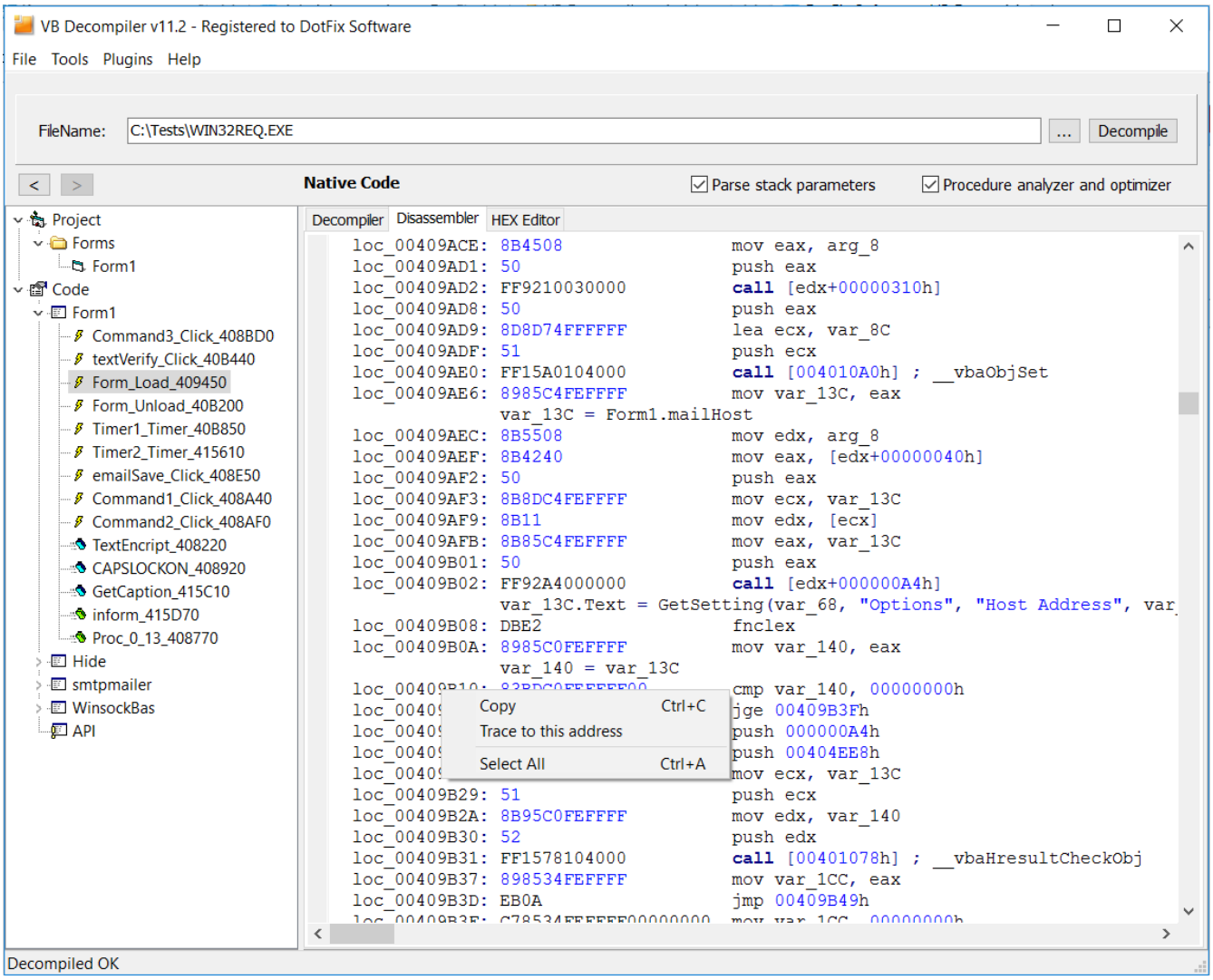
Bir izleme başlatma

1 Bir VB5/6 yordamını açın ve listede adresleri görünür tutun.

2 Araştırılacak satırda, kod bağlam menüsünden **Bu adresi takip et (Trace to this address)**'yi çağırın veya **F4**'ye basın.

3 İzleyicinin kod ve durum görüntülerini inceleyin ve **Adım (Next)** kullanarak ilerleyin. İzleme bittiğinde iptal edin.

26.4 sürümünden itibaren, komut, satırın başındaki boşlukları kesme işleminden sonra `loc_` adresini çözebilir. İmleci tam olarak adres etiketinde değil, o satırdaki başka bir yerde yerleştirebilirsiniz.



Yerel Kod durumu

İzleyici, emülasyon ilerledikçe CPU ve FPU yazmaçlarını, yığını ve değişkenleri ve yeniden yapılandırılmış kodu ortaya koyar. Bu görünümleri kullanarak bir değerin veya ifadenin nasıl türetilmiş olduğunu inceleyin. Değişken-listesi arama ve mevcut nesnenin kurtarılan tipinin gösterimi daha büyük yordamlar için yardımcı olabilir.

P-Kod izleme

11 sürümünde tanıtılan P-Kod izleme, değişken bilgisi ve geri derlenmiş ifadelerin işleme sırasında nasıl geliştiğini gösterir. Bu, ara P-Kod işlemleri ile optimize edilmiş yüksek seviye liste arasındaki farkları açıklamaya yardımcı olabilir.

Kapsam ve sınırlamalar

- İzleme tek bir yordam için gerçekleştirilir. Çalışan bir sürecin durumu yerine, öykünülen yazmaçların, yığının ve değişkenlerin boş durumuyla başlar.
- Desteğe alınmayan Yerel Kod komutları atlanabilir. Dış fonksiyon çağrıları çalıştırılmaz; desteklenen çağrı kurallarının işlenmesi, dış fonksiyonun davranışını yeniden üretmeden modellenen yığını ayarlayabilir.
- İzleyici, tam işlemci-bayraklarını göstermez. Durumunu tam donanım çalışma izi olarak yorumlamayın.
- Açıklanan izleme modeli, döngü iterasyonlarının tekrarlı çalıştırılması veya ters izleme olmadan ileri adım adım emülasyonudur.
- Bu iş akışı VB5/6 izlemeyi açıklar. .NET makine kodu çözümlemesinin bulunması, .NET çalışma zamanı hata ayıklayıcısı bulunduğu anlamına gelmez.

İzlemeyi, geri derleyicinin kodu nasıl yeniden oluşturduğunu ve ara değerleri incelemek için kullanın. Tek başına öykünülen bir değer; başlatılmış global değişkenlere, dış bağımlılıklara ve girdi verilerine sahip gerçek bir sürecin üreteceği değeri kanıtlamaz.

12 Analiz özellikleri

Analytic features lisansları, analiz edilen programda tanınan işlemlerin kategorize edilmiş raporlarını oluşturur. Bu raporlar, güvenlik uzmanlarının tüm kod tabanını okumadan önce incelemeye değer yordamları belirlemelerine yardımcı olur. Visual Basic 5.0/6.0 uygulamalarını ve 11.7 sürümünden itibaren .NET derlemelerini destekler.

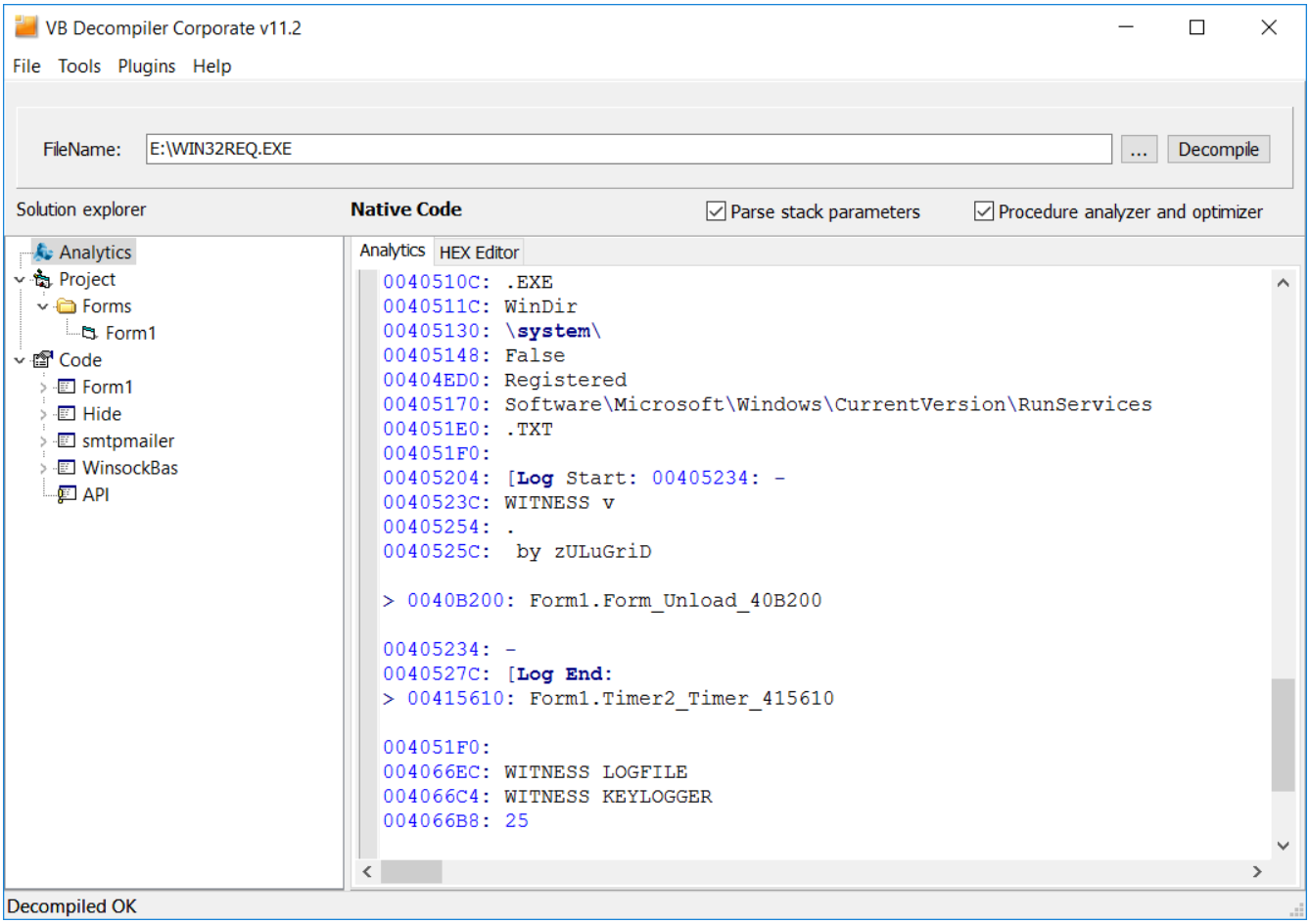
Rapor oluşturma

1. **Prototipleri analiz et (Analyze prototypes)**'yi **Araçlar > Ayarlar (Tools > Options)**'de etkinleştirin.
2. Dosyayı açın ve analizin tamamlanmasını bekleyin. Gerekli lisans ve ayarlarla rapor otomatik olarak oluşturulur ve görüntülenir.
3. Kategorileri gözden geçirin, ardından ilgili yordamları açın ve kodlarını ve string referanslarını inceleyin.

Kategoriler, dosya ve disk birimi erişimi, kayıt defteri erişimi, ağ aktivitesi, süreç oluşturma, pencere işlemleri ve seçili VB içsel fonksiyonları (örneğin `CallByName` ve `VarPtr`) gibi tanınan işlemleri kapsar.

Hızlı rapor ve string bağlamı

Hızlı rapor, tanınan işlemleri içeren yordamların adreslerini ve adlarını listeler. Genişletilmiş rapor, bu yordamlardan mevcut string referanslarını ekler. Bu bağlam, incelenmeye değer aday yolları, kayıt defteri anahtarlarını, sunucu adlarını veya diğer değerleri ortaya çıkarabilir.



Şifrelenmiş, kodu karartılmış veya çalışma zamanında üretilen dizeler eksik olabilir. Bir yordamle ilişkilendirilen bir string yararlı bağlam sağlar ancak tek başına belirli bir API çağrısına aktarıldığını veya bir dalın yürütüldüğünü kanıtlamaz.

Sonuçları gezinme ve kaydetme

Raporda bir yordam adını çift tıklayarak kodunu açabilirsiniz. Bir yordam veya string'den önceki adresi çift tıklayarak HEX Editörü (HEX Editor)'deki konumunu inceleyebilirsiniz. Önceki konuma dönmek için nesne ağacının üzerindeki **A** düğmesini kullanın ve incelenmiş yordamlara geri dönmek için gezinme oklarını veya **Esc**'yi kullanın.

Gerekirse rapor metnini seçip kopyalayın veya daha fazla analiz için kaydetmek üzere **Dosya > Analitik raporu kaydet (File > Save analytics report)**'yi seçin.

Bulguları yorumlama

Rapor, statik olarak tanınan kod desenlerini ve referanslarını belirler. Gözlemlenen yürütmenin kaydı veya bir dosyanın kötü amaçlı olduğuna dair otomatik bir hüküm veren bir araç değildir; ön incelemeye yardımcı olur. Dolaylı çağrılar, çözülmemiş dış arayüzler, hasarlı meta veriler, kod karartma ve çalışma zamanında üretilen değerler kapsamı sınırlayabilir. Önemli bulguları ilgili yordamların kodunda, makine kodu çözümlemesinde ve dosya verilerinde doğrulayın.

Daha derinlemesine VB5/6 yeniden inşası için İzleme'yı inceleyin. Dize sabitleri ve kod konumları için String Referansları'na bakın.

13 Kod karartma sınırlamaları

VB Decompiler, benzersiz bir kod karartma teknolojisiyle donatılmıştır ve **Visual Basic 5.0 ve 6.0 uygulamaları**'nin geri derlenmesindeki araştırmamızda derleyici tarafından geride bırakılan verileri etkili bir şekilde gizler. Modern programlama dilleri genellikle derlenmiş dosyaların içinde önemli miktarda meta veri ve hata ayıklama bilgisi saklar, bu da kırıcıların programınızın içinde istenen fonksiyonları bulmasını kolaylaştırır. Bu tehdide karşı, **Visual Basic** uygulamaları için özel olarak tasarlanmış bir kod karartma aracı geliştirdik.

Kod karartma işleminin ayrıntıları

Kod karartma aracı, programı formlar, modüller, sınıflar ve butonlar ile metin alanları gibi kontroller dahil olmak üzere bileşenlerine ayırarak çalışır. Her form daha sonra ilgili olaylarla birlikte bireysel kontroller halinde ayrıştırılır. Ardından, fonksiyonlar ve yordamlar hakkında bilgi toplanır ve programdaki her nesneye bağlı olaylar da derlenir. Bu süreçte, kod karartma aracı nesneler arası bağımlılıkları kontrol eder ve bir modüle veya sınıfa ait hangi fonksiyona ait olduğunu belirlemeyi zorlaştırmak için isimlerini senkron olarak değiştirir.

Sonuç olarak, programın içinde **frmTrial** veya **frmRegistration** gibi belirli formları bulmak önemli ölçüde zorlaşır.

Visual Basic 5.0/6.0 uygulamalarında kod karartma sınırlamaları

Kod karartma aracı tarafından kullanılan algoritma karmaşıktır ve nesne bağımlılıklarını analiz etmek üzere tasarlanmıştır, ancak hala her şeyi desteklememektedir. Bu nedenle bazı durumlarda, kod karartma sürecinden önce programınızı hazırlamanız gerekebilir. Aşağıda tartışılan noktalar yalnızca Visual Basic 5.0 / 6.0 uygulamalarının kod karartma işlemine uygulanır.

1. Kod karartma aracı, çalışma zamanı sorunlarına yol açabilecek diğer nesnelere tasarım zamanı referanslarını desteklemez.

Örneğin, **TreeView** bir kontrolü bağlı olan bir **ImageList**'ye bağımlı olduğunuz bir senaryoyu düşünün. Bu **ImageList** nesnesinin ismi, **TreeView**'nin özelliklerine sabitlenmişse, herhangi bir kod karartma girişimi hatalara yol açacaktır çünkü orijinal referans yerinde kalır. Bu kısıtlama ayrıca **DataSets** gibi veritabanı bileşenlerine de uygulanır.

```
Begin ComctlLib.TreeView TreeView1
  Height          =   1932
  Left            =   3600
  TabIndex        =   3
  Top             =   1320
  Width           =   2412
  _ExtentX        =   4255
  _ExtentY        =   3408
  _Version        =   327682
  Style           =   7
  ImageList       =   "ImageList1"
  Appearance      =   1
End
```

Bu sorun, bu atamaları **Form_Load** olayına taşıyarak tasarım zamanından çalışma zamanına taşıyarak kolayca çözülebilir:

```
Private Sub Form_Load()
  TreeView1.ImageList = ImageList1
End Sub
```

Aynı düzeltme, her aynı örnek için uygulanmalıdır. Neyse ki, çoğu projede genellikle bu tür birkaç durum vardır.

2. As New ile nesne bildirimleri desteklenmez (LateCall çağrıları)

Bu kısıtlama genellikle çoğu VB6 projesini etkilemez, ancak bunu aklınızda tutmanız önemlidir. Formları sınıflar olarak çalıştırıyor ve form örneklerini oluşturuyorsanız bu senaryoyu tanıyabilirsiniz:

```
Private Sub Form_Load()
  Dim oForm as New frmMain
  oForm.Show
End Sub
```

Bu tür durumlarda, VB6 derleyicisi **Form_Load** fonksiyonunun koduna doğrudan **frmMain** adını yerleştirir. Kod karartma aracımız yalnızca verileri değiştirir, kodu değiştirmez. Bu nedenle koda gömülü ad, kod karartma işleminden sonra da değişmeden kalır. Bu durum, programın hatayla sonlanmasına yol açar.

Ne yazık ki, **VB Decompiler**'nin kod karartma aracında bu tür çağrıları desteklememiz mümkün değildir çünkü tüm işlevlerin makine kodunu çözümlememiz ve geri derlememiz gerekir. Bu yeteneğe sahip olmasına rağmen, geri derleme işlemi son derece zaman alıcı bir süreçtir. Büyük projeler için tüm kod tabanını geri derlemek yarım saatten fazla sürebilir. Ayrıca, nesnelere referansları analiz etmeli ve isimlerini değiştirmeliyiz, bu da daha fazla işleme süresi ekler.

Bu teknik kısıtlamalar göz önüne alındığında, **VB Decompiler**'nin kod karartma aracında tam geri derleme ve kod analizi uygulamamaya karar verdik, özellikle bu durum Visual Basic 5.0 / 6.0 uygulamalarında çok nadir durumlarda sorunu çözeceği ve çoğu müşteri için herhangi bir fayda sağlamayacağı için.

14 Teknik destek

Programla ilgili sorularınız varsa ve bu yardım rehberinde cevap bulamazsanız, lütfen teknik destek ekibimizle iletişime geçin. Bu seçenek yalnızca kayıtlı kullanıcılar için mevcuttur. Kayıtlı olmayan kullanıcılar için **VB Decompiler** lisansı satın alma konularında sorularınıza memnuniyetle cevap veriyoruz.

Ayrıca, program içinde herhangi bir hata ile karşılaşırsanız veya onu geliştirmek için önerileriniz varsa, geri bildirimlerinizi teknik destek hizmetimize iletmekten çekinmeyin.

15 Kayıt

VB Decompiler'nin tam yeteneklerini kullanmak için **Business License** satın almanız gerekir. Tek Geliştirici İş Lisansı veya Kurumsal Lisans satın alarak şu yetenekleri elde edersiniz:

1. Desteklenen tipik **Visual Basic 5.0/6.0 P-Code** uygulamalarında kodun yaklaşık %85'ini kurtarma;
2. Desteklenen tipik **Visual Basic 5.0/6.0 Native Code** uygulamalarında kodun %85'ine kadarını kurtarma;
3. Desteklenen, kod karartma uygulanmamış tipik **.NET** derlemelerini **C#** benzeri koda geri derleme; %95'e kadar kurtarma;
4. Programın en son sürümüne sahip olma: Lisans satın aldıktan sonra bir yıl boyunca güncellemeler tamamen ücretsizdir;
5. Lisans satın aldıktan sonra bir yıl boyunca e-posta yoluyla teknik destek alma;
6. Bu decompiler'ın geliştirilmesini destekleme.

Ayrıca, Malware Analistleri ve dijital adli bilişim uzmanları için özel lisansların satın alınmasını öneriyoruz:

1. tracing özelliklerine sahip lisans
2. analitik özelliklere sahip lisans

Geri derleyici için bir **lisans** satın almak üzere programın **Yardım (Help)** menüsünde **Pro versiyonu sipariş et (Order Pro version)** seçeneğini seçin. Bu anda internete bağlı olmalısınız. Tarayıcı penceresi, lisansı alabileceğiniz programın web sitesine açılacaktır.

Decompiler'ın **Business License**'si yalnızca bir bilgisayarda ve yalnızca bir **Single Developer** (lisans sahibi) tarafından kullanılacak şekilde tasarlanmıştır. Programı birden fazla bilgisayarda kullanmanız gerekiyorsa, her biri için ayrı bir lisans satın almanız veya **Business Enterprise license**'i (en fazla 20 bilgisayarda kullanılabilir) satın almayı düşünmelisiniz.

16 Geliştirici hakkında

Bu geri derleyici **DotFix Software** tarafından geliştirilmektedir.

Ürün web sitesi: www.vb-decompiler.org
Teknik destek e-postası: admin@vb-decompiler.org