



**T.C.
KIRKLARELİ ÜNİVERSİTESİ
TEKNİK BİLİMLER MESLEK YÜKSEKOKULU
BİLGİSAYAR TEKNOLOJİLERİ BÖLÜMÜ
BİLGİSAYAR PROGRAMCILIĞI PROGRAMI**

PYTHON: KIVY KÜTÜPHANESİ

CEREN YALNIZ

1247008014

PYTHON

NADİR SUBAŞI

KIRKLARELİ

13.12.2025

İÇİNDEKİLER

KISALTMALAR	2
PYTHON KIVY KÜTÜPHANESİ.....	4
Kivy Nedir?	4
Kim Tarafından Geliştirilmektedir?	4
Lisansı Nedir?	5
Çözdüğü Problem	5
Kivy Neden Ortaya Çıktı?	5
a.) Mobil Platformlarda Çalışmaması	6
b.) Dokunmatik Ekranlara Uygun Olmaması	6
c.) Modern Tasarım Trendlerinden Geri Kalması	6
d.) OpenGL Tabanlı Yüksek Grafik Performansı Sunmaması	6
e.) Mobil Paketleme Sorunları	7
Kivy'nin Karşılıdığı Spesifik İhtiyaçlar	7
1. Tek Kod Tabanı ile Çoklu Platform Desteği	7
2. Dokunma ve Multi-Touch Desteği	8
3. GPU Hızlandırmalı Modern Arayüz	8
4. Python ile Mobil Uygulama Geliştirebilme	8
5. Modern ve Esnek Arayüz Tasarımına İzin Vermesi	9
Temel Özellikleri.....	9
1 Cross-Platform Yapı	9
2 OpenGL ES 2 Tabanlı Grafik Motoru.....	9
3 Multi-Touch ve Gesture Desteği	9
4 KV Language ile Tasarım – Mantık Ayrımı.....	9
5 Hazır Widget ve Layout Desteği	10
Kurulum ve Bağımlılıklar	10
1. Kivy'nin Kurulumu	10
2. Gerekli Bağımlılıklar	10
3. Mobil Platformlar İçin Ek Araçlar	11
4. Ek Kütüphaneler	11
Ekosistemdeki Yeri ve Alternatifleri	11
Kullanım Alanları (Case Study)	12

Sonu	13
Kaynaka	14

KISALTMALAR

Bu raporda kullanılan teknik terimler ve kısaltmalar aşağıda açıklanmıştır:

API – Application Programming Interface (Uygulama Programlama Arayüzü)

CPU – Central Processing Unit (Merkezi İşlem Birimi)

GPU – Graphics Processing Unit (Grafik İşlem Birimi)

GUI – Graphical User Interface (Grafiksel Kullanıcı Arayüzü)

KV – Kivy Language (Kivy Arayüz Tanımlama Dili)

OpenGL – Open Graphics Library

SDK – Software Development Kit (Yazılım Geliştirme Kiti)

NDK – Native Development Kit

APK – Android Application Package

IPA – iOS Application Archive

FPS – Frames Per Second (Saniyedeki Kare Sayısı)

MIT – Massachusetts Institute of Technology (MIT Açık Kaynak Lisansı)

PYTHON KIVY KÜTÜPHANESİ

Günümüzde yazılım geliştirme ekosistemi, çoklu platformlarda çalışabilen uygulamalar geliştirmeye yönelmektedir. Mobil cihazların yaygınlaşması, dokunmatik ekranlı sistemlerin artması ve kullanıcıların platform bağımsız çalışan yazılımlara ihtiyaç duyması, geliştiricileri bu yönde araçlar kullanmaya itmektedir. Python programlama dili, kullanım kolaylığı, geniş kütüphane desteği ve hızlı prototip geliştirme olanakları sayesinde dünya genelinde en popüler dillerden biri olmuştur. Ancak Python'un standart kütüphaneleri, özellikle mobil uygulama geliştirme konusunda yeterli değildir.

Bu nedenle Python geliştiricilerinin mobil ve masaüstü için modern, dokunmatik destekli arayüzler oluşturabilmesi adına özel framework'lere ihtiyaç duyulmuştur. Bu ihtiyacı karşılayan en güçlü araçlardan biri Kivy kütüphanesidir. Kivy, Python diliyle hem masaüstü hem de mobil platformlarda çalışan, modern, animasyonlu, çoklu dokunma destekli kullanıcı arayüzleri geliştirmeye olanak sağlayan açık kaynaklı bir framework'tür.

Bu raporda Kivy kütüphanesi kapsamlı bir şekilde incelenecek; tanımı, ortaya çıkış amacı, özellikleri, kurulumu, ekosistemi, alternatifleri ve gerçek dünya kullanım örnekleri akademik bir dille açıklanacaktır.

Kivy Nedir?

Kivy, Python ile geliştirilmiş, çoklu platform desteğine sahip bir arayüz geliştirme framework'üdür. Tek bir kod tabanı kullanarak Windows, Linux, macOS, Android ve iOS üzerinde çalışan uygulamalar geliştirmeyi mümkün kılar. Dokunmatik ekran desteği, GPU hızlandırılmış grafik motoru ve modern widget yapıları sayesinde klasik GUI kütüphanelerinden ayrılır.

Kim Tarafından Geliştirilmektedir?

Kivy, ilk olarak 2010 yılında Mathieu Virbel tarafından geliştirilmeye başlanmıştır. Günümüzde geliştirme sürecini: Kivy Organization adı verilen bir açık kaynak topluluğu, dünya çapında yüzlerce gönüllü yazılımcı, gitHub üzerinden katkı sağlayan bağımsız geliştiriciler sürdürmektedir.

Kivy tamamen açık kaynaklı bir projedir. Yani herkes projeye katkı sağlayabilir, kodu inceleyebilir, değişiklik önerisi sunabilir veya kendi uygulamalarında özgürce kullanabilir.

Lisansı Nedir?

Kivy, MIT Lisansı ile lisanslanmıştır ve bu lisans, açık kaynak dünyasında en esnek ve kullanıcı dostu lisanslardan biri olarak kabul edilmektedir. MIT lisansı sayesinde Kivy; ticari projelerde kullanılabilir, kaynak kodu değiştirme ve yeniden dağıtma özgürlüğü sunar ve geliştiricilere kaynak kodu açık tutma zorunluluğu getirmez. Bu özellikler, Kivy'nin eğitim, akademik, kişisel ve kurumsal projelerde herhangi bir kısıtlama olmadan kullanılabilmesini mümkün kılar. Sağladığı bu özgürlük ortamı sayesinde Kivy, özellikle mobil uygulama geliştirmek isteyen öğrenciler, girişimciler ve küçük yazılım ekipleri tarafından sıkça tercih edilmektedir.

Kivy'nin bu noktada öne çıkmasının temel nedenlerinden biri, klasik Python GUI kütüphanelerinin sahip olduğu sınırlamaları aşabilmesidir. Geleneksel arayüz araçları genellikle mobil cihazlarda çalışmaz, dokunmatik ekranlara uygun değildir, OpenGL tabanlı yüksek grafik performansı sunamaz ve modern animasyon ile tasarım trendlerine yeterince uyum sağlayamaz. Bu durum, Python geliştiricilerinin mobil uygulama geliştirmek istediklerinde Java veya Kotlin (Android) ile Swift veya Objective-C (iOS) gibi farklı dilleri öğrenmek zorunda kalmasına yol açmıştır. Kivy, sunduğu esnek lisans yapısı ve teknik yetenekleriyle bu ihtiyaca doğrudan çözüm getiren bir framework olarak ortaya çıkmıştır.

Çözdüğü Problem

Kivy'nin ortaya çıkış amacı, Python ekosisteminde uzun süredir var olan bir boşluğu doldurmaktır. Python, güçlü ve çok amaçlı bir dil olmasına rağmen, özellikle mobil uygulama geliştirme ve dokunmatik arayüzlere sahip modern grafiksel kullanıcı arayüzleri (GUI) konusunda yetersiz bir altyapıya sahipti. Klasik GUI kütüphaneleri yalnızca masaüstü odaklıydı ve mobil cihazların artan kullanımına cevap veremiyordu. Bu nedenle Kivy, geliştiricilerin hem mobil hem masaüstü platformlar için aynı kod tabanıyla çalışabilen modern arayüzler geliştirebilmesini sağlamak amacıyla tasarlanmıştır.

Kivy Neden Ortaya Çıktı?

Python ile GUI uygulamaları geliştirmek uzun zamandır mümkündür, ancak bu alanda kullanılan araçlar ağırlıklı olarak masaüstü sistemler için tasarlanmıştı. Bu klasik kütüphaneler şunlardır:

- **Tkinter:** Python'un standart GUI kütüphanesidir. Basit masaüstü uygulamaları için uygundur ancak modern tasarım trendlerini desteklemez.
- **PyQt / PySide:** Qt framework'ünün Python bağlayıcılarıdır. Güçlü ve modern masaüstü uygulamaları geliştirmek mümkündür, ancak mobil cihazlara resmi ve kolay bir şekilde paketlemek zordur.

- **wxPython:** Platforma özgü bileşenler kullanır, bu nedenle masaüstünde iyi çalışır, ancak mobil ortamlara uygun değildir.

Bu araçların ortak sınırlamaları şunlardır:

a.) Mobil Platformlarda Çalışmaması

Klasik GUI kütüphaneleri mobil cihazlar için tasarlanmadığından Android veya iOS üzerinde çalışmaz. Mobil ekran boyutlarına uyum sağlayamaz ve dokunmatik girdileri algılayamazlar. Bu nedenle Python geliştiricileri mobil uygulama geliştirmek için başka dillere yönelmek zorunda kalmıştır.

b.) Dokunmatik Ekranlara Uygun Olmaması

Günümüzde hem mobil cihazlarda hem de bazı masaüstü bilgisayarlarda dokunmatik ekranlar sıkça kullanılmaktadır. Ancak klasik GUI kütüphaneleri, aynı anda birden fazla dokunuşu algılama, yakınlaştırma, kaydırma veya döndürme gibi temel dokunmatik hareketleri desteklemez. Bu nedenle dokunmatik etkileşimlere dayalı uygulamalar geliştirirken önemli sınırlamalar ortaya çıkar. Özellikle oyunlar, fotoğraf galerileri ve eğitim uygulamaları gibi dokunma odaklı arayüzlere ihtiyaç duyan projeleri oluşturmak oldukça zor hale geliyordu..

c.) Modern Tasarım Trendlerinden Geri Kalması

Son yıllarda kullanıcılar, akıcı animasyonlara, yumuşak geçişlere, modern ikonlara ve kart tabanlı düzenlere sahip dinamik arayüzlere alışmış durumda. Ancak klasik GUI kütüphaneleri bu tasarım anlayışını karşılamakta yetersiz kalabiliyor. Tkinter, görünüm olarak oldukça eski bir yapıya sahipken; PyQt, geniş imkânlar sunmasına rağmen animasyon ve modern etkileşimlerde yeterli esnekliği sağlayamıyor. Bu durum, çağdaş tasarım beklentilerine uygun arayüzler geliştirmeyi bu araçlarla zorlaştırıyor.

d.) OpenGL Tabanlı Yüksek Grafik Performansı Sunmaması

Pek çok uygulamada arayüzün akıcı çalışması ve animasyonların sorunsuz görünmesi için güçlü bir grafik altyapısı gerekir. Ancak klasik GUI kütüphaneleri, grafik işlemlerini çoğunlukla CPU üzerinden gerçekleştirdiği için animasyonların yavaşlaması, geçişlerin ağırlaşması ve oyun benzeri uygulamalarda düşük kare hızları gibi sorunlar ortaya çıkabilir. CPU'nun bu yükü tek başına karşılamaya çalışması performansı düşürür ve görsel açıdan zengin uygulamalarda belirgin kısıtlamalara yol açar. Oysa grafik işlemlerinin GPU tarafından işlenmesi, hem hız hem de verimlilik açısından çok daha başarılı sonuçlar üretir.

e.) Mobil Paketleme Sorunları

Python ile yazılmış bir uygulamayı mobil cihazlarda çalıştırmak, uzun süre geliştiriciler için önemli bir engeldi. Android tarafında Java ya da Kotlin, iOS tarafında ise Swift veya Objective-C kullanma zorunluluğu, Python bilgisiyle mobil uygulama geliştirmeyi neredeyse imkânsız hâle getiriyordu. Üstelik Python kodunu doğrudan bir APK veya IPA dosyasına dönüştürebilecek pratik bir yöntem de bulunmadığından, geliştiriciler farklı platformlar için farklı diller öğrenmek zorunda kalıyordu. Bu durum hem süreci karmaşıktırıyor hem de Python’ın mobil geliştirme alanında geri planda kalmasına neden oluyordu. İşte tam bu noktada Kivy devreye girerek Python projelerinin mobil cihazlarda da çalışabilmesini sağlayan, bu açığı kapatan önemli bir çözüm olarak ortaya çıkmıştır.

Kivy’nin Karşıladığı Spesifik İhtiyaçlar

Kivy, Python dünyasındaki önemli problemlere bütünsel çözümler sunar. Bu çözümler aşağıda detaylı şekilde açıklanmıştır.

1. Tek Kod Tabanı ile Çoklu Platform Desteği

Farklı platformlar için farklı programlama dillerinde uygulama geliştirmek, hem öğrenme maliyetini hem de geliştirme sürecinin zorluk derecesini önemli ölçüde artırmaktadır. Örneğin Android için Java veya Kotlin, iOS için Swift ya da Objective-C, masaüstü ortamları için ise bambaşka araç ve teknolojiler öğrenmek gerekir. Bu durum, özellikle küçük yazılım ekiplerinin iş yükünü ağırlaştırmakta ve projelerin sürdürülebilirliğini zorlaştırmaktadır. Her platform için ayrı kod tabanı oluşturmak, hem geliştirme sürecini uzatmakta hem de bakım sürecini karmaşık bir hâle getirmektedir. Kivy ise bu çok parçalı yapıyı ortadan kaldırarak geliştiricilere büyük bir kolaylık sunar. Kivy ile uygulama geliştirmek için yalnızca Python bilmek yeterlidir; geliştirici tek bir kod tabanı oluşturur ve bu kod, ek bir dil öğrenmeden farklı platformlara paketlenir. Arayüz tasarımları da Kivy’nin esnek yapısı sayesinde tüm cihaz türlerine uyum sağlayacak biçimde hazırlanabilir. Bu yaklaşım, geliştirme sürecine hem pratiklik hem de hız kazandırır. Zaman tasarrufu sağlar, küçük ekiplerin veya tek geliştiricilerin bile çoklu platformda uygulama yayınlamasını mümkün kılar ve prototip geliştirme sürecini büyük ölçüde hızlandırır. Bu nedenle Kivy, özellikle öğrenciler, yeni başlayanlar ve bireysel geliştiriciler için ideal bir çalışma ortamı oluşturur. Kısacası Kivy, “bir kez yaz, her yerde çalıştır” anlayışını Python ekosisteminde gerçeğe dönüştüren güçlü bir araçtır.

2. Dokunma ve Multi-Touch Desteği

Kivy’nin en dikkat çekici ve güçlü yönlerinden biri, dokunmatik girişleri doğal bir şekilde desteklemesidir. Geleneksel GUI kütüphaneleri genellikle fare ve klavye etkileşimleri üzerine kuruludur ve dokunma hareketlerini algılamak için ek eklentilere veya özel çözümlere ihtiyaç duyar. Oysa Kivy, modern cihazların gereksinimlerini göz önünde bulundurarak tasarlanmış olup tek dokunuş, çift dokunuş, uzun basma, sürükleme, pencere kaydırma gibi temel

etkileşimlerin yanı sıra iki parmakla yakınlaştırma (pinch-zoom), döndürme gibi gelişmiş jestleri ve aynı anda çoklu dokunmaları (multi-touch) algılayabilmektedir. Kivy'nin bu yetenekleri, özellikle mobil oyunlar, harita tabanlı uygulamalar, fotoğraf görüntüleyiciler, eğitim amaçlı etkileşimli uygulamalar ve kiosk sistemleri gibi dokunmatik arayüze ihtiyaç duyan projelerde büyük avantaj sağlar. Bu çok yönlü dokunma desteği sayesinde geliştiriciler, modern kullanıcı alışkanlıklarına uygun, sezgisel ve dinamik arayüzler oluşturabilir. Kivy'nin dokunmatik işlevleri çekirdek yapısına doğal olarak entegre edilmiştir; böylece geliştirici ek bir yapı kurmak zorunda kalmadan doğrudan çoklu dokunma özelliklerinden yararlanabilir.

3. GPU Hızlandırmalı Modern Arayüz

Kivy'nin mimarisini diğer pek çok GUI kütüphanesinden ayıran en önemli özelliklerden biri, grafik işlemlerini geleneksel yöntemlerde olduğu gibi CPU üzerinde değil, doğrudan GPU üzerinde gerçekleştirmesidir. OpenGL ES 2 tabanlı bu modern ve yüksek performanslı yaklaşım, uygulama arayüzünün daha yüksek kare hızında çalışmasını sağlayarak animasyonların son derece akıcı, kullanıcı deneyiminin ise doğal ve kesintisiz olmasına imkân tanır. GPU'nun işlem gücünden yararlanılması, grafiklerin gerçek zamanlı ve yüksek verimlilikle çizilmesini mümkün kılarken aynı zamanda CPU üzerindeki yükü büyük ölçüde azaltır. Bu durum yalnızca genel performans artışı sağlamakla kalmaz; aynı zamanda oyun benzeri interaktif uygulamaların, gelişmiş shader veya particle effect gibi hareketli grafik öğelerinin kullanılmasına da olanak tanır. Kivy'nin GPU hızlandırmalı yapısı sayesinde büyük ekran çözünürlüklerinde bile performans kaybı yaşanmaz ve arayüz karmaşık görsel yapıların altında dahi stabil şekilde çalışır. Tüm bu nitelikler, Kivy'yi klasik masaüstü araçlarının çok ötesine taşıyarak modern, dinamik ve yüksek görsel kaliteye sahip uygulamalar geliştirmek isteyenler için güçlü ve çağdaş bir çözüm hâline getirir.

4. Python ile Mobil Uygulama Geliştirebilme

Mobil uygulama geliştirmek geleneksel olarak Android için Java/Kotlin, iOS için ise Swift veya Objective-C gibi farklı dillerin öğrenilmesini gerektirir. Bu durum özellikle öğrenciler ve bireysel geliştiriciler için önemli bir öğrenme yükü oluşturur. Kivy, bu zorluğu ortadan kaldırarak yalnızca Python bilgisiyle her iki platform için de uygulama geliştirmeyi mümkün kılar. Bu sayede öğrenme süreci kısalır, daha hızlı sonuç elde edilir ve prototip geliştirme aşaması büyük ölçüde hızlanır. Mobil uygulama geliştirme sürecinin herkes için erişilebilir hâle gelmesini sağlayan Kivy, Android tarafında Buildozer aracıyla Python kodlarını APK formatına dönüştürerek geliştiricilere pratik ve güçlü bir çözüm sunar.

5. Modern ve Esnek Arayüz Tasarımına İzin Vermesi

Kivy, modern ve esnek arayüz tasarımlarını desteklemek amacıyla KV Language adı verilen deklaratif bir arayüz dili sunar. Bu özel dil sayesinde kullanıcı arayüzü, Python kodundan ayrı bir yapıda tanımlanabilir; böylece geliştirme süreci daha düzenli, okunabilir ve yönetilebilir hâle gelir. KV dili, widget'ların görünümünü ve davranışlarını açık ve sade bir söz dizimiyle tanımlamaya olanak tanıdığı için arayüzler çok daha hızlı ve sistematik bir şekilde oluşturulabilir. Bu yaklaşım, yapısal olarak Flutter'daki widget sistemine veya web geliştirmedeki HTML–CSS ayrımına benzer bir mimari sunar. Sonuç olarak geliştiriciler, hem modern tasarım trendleriyle uyumlu hem de profesyonel bir görünüm sağlayan kullanıcı arayüzlerini kolaylıkla tasarlayabilir.

Temel Özellikleri

1 Cross-Platform Yapı

Kivy, tek bir Python kod tabanıyla farklı işletim sistemlerinde çalışan uygulamalar geliştirmeye olanak tanır. Windows, Linux, macOS, Android ve iOS desteği sayesinde geliştiriciler, aynı uygulamayı birden fazla cihazda hızlıca test edip dağıtabilir. Bu yapı, özellikle küçük ekipler ve öğrenciler için büyük bir kolaylık sağlar.

2 OpenGL ES 2 Tabanlı Grafik Motoru

Kivy'nin grafik yapısı GPU hızlandırmalı OpenGL ES 2 üzerine kuruludur. Bu sayede arayüzler daha akıcı çalışır, animasyonlar yüksek performansla işlenir ve genel kullanıcı deneyimi modern bir görünüm kazanır. Bu yaklaşım Kivy'yi klasik GUI kütüphanelerine kıyasla daha hızlı ve dinamik bir yapı hâline getirir.

3 Multi-Touch ve Gesture Desteği

Kivy, çoklu dokunmayı doğal olarak destekleyen Python kütüphanelerinden biridir. Çift tıklama, uzun basma, kaydırma, döndürme ve pinch gibi birçok jest türü yerleşik olarak algılanabilir. Bu özellik, özellikle mobil ve dokunmatik ekranlı uygulamalarda Kivy'yi güçlü bir seçenek yapar.

4 KV Language ile Tasarım – Mantık Ayrımı

Kivy, arayüz tasarımı için KV Language adlı özel bir dil sunar. Bu yaklaşım:

- Arayüz ile uygulama mantığının ayrılmasını,
- Daha düzenli kod yapısını,
- XML benzeri okunabilir bir tasarım söz dizimini

mümkün kılar. Bu model HTML–CSS ile JavaScript ayrımına benzetilebilir ve geliştirme sürecini hızlandırır.

5 Hazır Widget ve Layout Desteđi

Kivy, modern bir arayüz geliřtirmek için birçok hazır bileřen sunar.

Widget Örnekleri: Button, Label, TextInput, Checkbox, Slider, Switch, DropDown, ProgressBar, ScreenManager

Layout Yapıları: BoxLayout, GridLayout, FloatLayout, AnchorLayout, RelativeLayout

Bu bileřenler, karmařık arayüzlerin kısa sürede oluşturulmasını saęlar.

Kurulum ve Baęımlılıklar

Kivy, Python tabanlı bir framework olduęu için kurulumu oldukça kolaydır. Geliřtiricilerin yalnızca uygun bir Python ortamına sahip olması ve gerekli paketleri yüklemesi yeterlidir. Kivy'nin farklı platformlarda sorunsuz çalışabilmesi için bazı baęımlılıklar bulunmakla birlikte, çoęu işlem pip aracılığıyla otomatik olarak gerçekleştirilir.

1. Kivy'nin Kurulumu

Kivy, Python ortamına ” pip install kivy” komutu ile yüklenebilir. Bu komut, Kivy'nin temel bileřenlerini ve gerekli alt paketlerini otomatik olarak indirir. Geliřtirme sürecinde bir sanal ortam (virtual environment) kullanılması önerilir; böylece proje baęımlılıkları dięer Python projelerinden izole şekilde yönetilebilir.

2. Gerekli Baęımlılıklar

Kivy farklı platformlarda çalıştığı için bazı ek baęımlılıklara ihtiyaç duyar:

OpenGL Desteđi

Kivy'nin grafik motoru GPU tabanlı çalıştığından, sistemde OpenGL desteęinin bulunması gerekir. Modern bilgisayarların ve mobil cihazların büyük çoęunluęu bu desteęi doęal olarak saęlar.

Cython

Cython, Python kodlarının C dili ile derlenerek çalıştırılmasını saęlayan ve bu sayede uygulamaların performansını artıran bir kütüphanedir. Kivy, özellikle grafik işlemleri gibi yüksek performans gerektiren bazı modüllerde daha hızlı sonuç alabilmek için Cython'dan yararlanır. Birçok sistemde Cython otomatik olarak yüklense de ihtiyaç duyulan durumlarda manuel olarak ” pip install cython“ komutu kullanılarak kurulum yapılabilir.

3. Mobil Platformlar İçin Ek Araçlar

Kivy ile geliştirilen bir uygulamanın mobil cihazlarda çalışabilmesi için, projenin her platforma uygun şekilde paketlenmesi gerekir. Bu nedenle Kivy, Android ve iOS için farklı araçlarla birlikte çalışır. Android tarafında en çok kullanılan araç **Buildozer**'dir. Buildozer, Python ile yazılmış bir Kivy uygulamasını Android işletim sistemi tarafından çalıştırılabilir hâle getiren bir derleyicidir ve bu işlem sonucunda bir APK dosyası oluşturur. Araç, yalnızca Linux üzerinde çalıştığı için geliştiricilerin bu işlemi Linux ortamında gerçekleştirmesi gerekir. Kurulum "**pip install buildozer**"komutu ile yapılır; ardından "**buildozer init**"komutu ile proje ayar dosyası oluşturulur ve "**buildozer -v android debug**"komutu kullanılarak uygulama derlenebilir. Derleme esnasında Buildozer, Android için gerekli SDK, NDK ve diğer bileşenleri otomatik olarak indirerek süreci kolaylaştırır.

iOS için Kivy uygulamalarını paketleme süreci daha farklıdır. Apple ekosistemi kapalı bir yapı olduğu için, iOS uygulamaları yalnızca macOS işletim sistemi ve Apple'ın geliştirme aracı olan Xcode kullanılarak derlenebilir. iOS SDK, sertifikalar ve Xcode derleyicisi bu sürecin ayrılmaz parçalarıdır. Kivy, projelerin Xcode içine aktarılmasına ve buradan bir iOS uygulamasına dönüştürülmesine olanak tanır. Bu sayede Python bilgisi olan geliştiriciler, hem Android hem de iOS platformları için mobil uygulama oluşturabilir ve her iki ekosistemde de çalışma imkânı elde eder.

4. Ek Kütüphaneler

Kivy uygulamalarını daha modern, işlevsel ve kullanıcı dostu hâle getirmek amacıyla çeşitli yardımcı kütüphanelerden yararlanılabilir. Bu kütüphanelerden **KivyMD**, Material Design prensiplerini temel alarak butonlar, menüler, kart yapıları ve diğer modern arayüz bileşenlerini hazır şekilde sunar ve kullanıcı deneyimini önemli ölçüde iyileştirir. **Plyer** ise kamera, GPS, titreşim ve sensörler gibi mobil cihaz donanımlarına erişim sağlayarak Kivy projelerinin gerçek cihaz özellikleriyle etkileşim kurmasını mümkün kılar. Daha teknik uygulamalarda kullanılan **OpenCV** gibi kütüphaneler de görüntü işleme, yüz algılama ve video analiz gibi ileri seviye işlevlerin Kivy projelerine entegre edilmesini sağlar. Tüm bu ek araçlar sayesinde Kivy ile geliştirilen uygulamalar hem görsel açıdan hem de teknik kapasite bakımından çok daha zengin ve kapsamlı bir kullanıcı deneyimi sunabilir.

Ekosistemdeki Yeri ve Alternatifleri

Kivy, Python ekosisteminde modern kullanıcı arayüzleri geliştirmeye yönelik sunduğu kapsamlı özellikleriyle stratejik bir konuma sahiptir. Python'un standart GUI araçları ağırlıklı olarak masaüstü platformlarını hedeflerken, Kivy hem masaüstü hem de mobil cihazlarda çalışabilen, dokunmatik ekran etkileşimlerini doğal olarak destekleyen ve GPU hızlandırmalı grafik yapısıyla öne çıkan bir framework'tür. Bu yönüyle, özellikle mobil uygulama geliştirme alanında Python dünyasında yer alan büyük bir boşluğu doldurur. Açık kaynak topluluğunun aktif desteği, düzenli güncellemeler ve geniş bir kullanıcı kitlesi sayesinde Kivy, eğitim kurumlarında, prototip çalışmalarında, deneysel projelerde ve bağımsız

geliştiriciler tarafından sıklıkla tercih edilen esnek bir araç hâline gelmiştir. Geliştiricilere hem hızlı prototipleme hem de modern tasarım imkânı sunması, Kivy'nin ekosistemdeki önemini daha da artırmaktadır.

Kivy'ye alternatif olabilecek diğer Python GUI kütüphaneleri incelendiğinde **PyQt/PySide** öne çıkan seçeneklerden biridir. Bu araçlar, Qt framework'ünün sunduğu geniş widget yelpazesi sayesinde kurumsal seviyede profesyonel masaüstü uygulamaları oluşturmayı mümkün kılar. Native bir masaüstü görünümü sunmaları, güçlü bileşen yapıları ve geniş dokümantasyon desteği ile masaüstü geliştirme odağında oldukça güçlüdürler. Ancak mobil platformlar için resmi ve pratik bir destek sunmamaları, onları çoklu platform geliştirme ihtiyacı bulunan projelerde Kivy'nin gerisinde bırakmaktadır. Ayrıca PyQt'nin ticari projelerde lisans gerektirmesi, açık kaynak ve özgür lisanslama avantajına sahip olan Kivy ile kıyaslandığında önemli bir dezavantaj olarak ortaya çıkar.

Bir diğer alternatif olan **Tkinter**, Python ile birlikte gelen standart arayüz kütüphanesidir ve basit masaüstü uygulamalarının hızlı bir şekilde geliştirilmesine uygundur. Öğrenmesi kolaydır ve ek bir kurulum gerektirmez. Buna rağmen, Tkinter modern kullanıcı arayüzü tasarımlarını desteklemez, animasyon kapasitesi sınırlıdır ve mobil platformlara yönelik hiçbir yetenek sunmaz. Bu yönleriyle, çağdaş uygulama geliştirme gereksinimlerinin çok gerisinde kalmaktadır ve özellikle dokunmatik cihazlara yönelik uygulamalar için uygun değildir.

Bu çerçevede değerlendirildiğinde, Kivy'nin konumu daha net ortaya çıkmaktadır: PyQt/PySide profesyonel masaüstü yazılımları için güçlü bir seçenek sunarken, Tkinter temel masaüstü projelerine yönelik en basit çözümlerden biridir. Kivy ise hem mobil hem masaüstü platformları kapsayan geniş kullanım alanı, dokunmatik ekran desteği, GPU hızlandırmalı grafik motoru ve modern tasarım araçlarıyla Python ekosisteminde benzersiz bir yere sahiptir. Özellikle çoklu platform desteğine ihtiyaç duyan, modern arayüz tasarımını önemseyen ve Python bilgisiyle mobil uygulama geliştirmek isteyen kullanıcılar için Kivy, mevcut alternatifler arasında en kapsamlı ve çağdaş çözüm olarak öne çıkar.

Kullanım Alanları (Case Study)

Kivy, çoklu platform desteği, dokunmatik ekran uyumluluğu ve GPU hızlandırmalı yapısı sayesinde hem eğitim hem oyun geliştirme hem de prototipleme alanlarında tercih edilen bir framework'tür. Python tabanlı oluşu, özellikle hızlı uygulama geliştirmek isteyen bireysel geliştiricilere ve eğitim kurumlarına pratik bir çözüm sunar.

Kivy'nin yeteneklerini en kapsamlı biçimde gösteren örneklerden biri, geliştirici topluluğu tarafından hazırlanan Kivy Showcase App'tir. Google Play ve App Store üzerinden erişilebilen bu uygulama, Kivy'nin sunduğu widget'lar, animasyonlar, geçiş efektleri ve dokunma hareketlerini tek bir uygulamada sergileyerek framework'ün uygulanabilirliğini açıkça ortaya koyar. Showcase uygulaması aynı zamanda geliştiriciler için referans niteliği taşıyan bir rehber işlevi görür.

Kivy'nin pratik kullanımını gösteren bir diğ er  rnek, pop ler bulmaca oyunu 2048'in Kivy ile geliřtirilmiř mobil s r m d r. Topluluk tarafından oluřturulan bu  alıřma, Kivy'nin oyun geliřtirme a ısından yeterli performansı saėlayabildiėini kanıtlar. GPU tabanlı grafik motoru sayesinde oyun akıcı bir deneyim sunar ve  oklu dokunma desteėi,  zellikle mobil kullanıcılar i in sezgisel bir oynanıř saėlar.

Eėitim alanında da Kivy'ye y nelik  eřitli projeler geliřtirilmiřtir. Edukivy adıyla bilinen uygulamalar,  zellikle matematik, řekil tanıma ve temel biliřsel beceriler  zerine hazırlanan etkileřimli i erikler sunar. Dokunma hareketlerini algılama, animasyon desteėi ve farklı ekran boyutlarına uyum saėlama gibi  zellikler sayesinde Kivy, eėitim materyallerinin daha etkileřimli ve ilgi  ekici h le gelmesine yardımcı olur.

Bu  rnekler, Kivy'nin yalnızca basit aray zlerin  tesine ge erek oyun, eėitim ve  oklu platform uygulamalarına kadar uzanan geniř bir kullanım alanına sahip olduėunu g stermektedir. Esnek mimarisi ve Python'un saėladığı kolaylıklar, Kivy'yi modern uygulama geliřtirmek isteyenler i in etkili bir ara  h line getirir.

Sonuç

Kivy, Python ile modern ve  ok y nl  uygulamalar geliřtirmek isteyenler i in  nemli bir bořluėu dolduran bir framework olarak  ne  ıkmaktadır. Geleneksel GUI k t phanelerinin mobil cihazlarda  alıřmaması, dokunmatik etkileřimleri desteklememesi ve g ncel tasarım beklentilerinin gerisinde kalması, Python geliřtiricilerinin hareket alanını uzun s re kısıtlamıřtı. Kivy ise hem masa st  hem de mobil platformlarda  alıřabilen yapısı, GPU hızlandırmalı grafik motoru ve dokunmatik ekran desteėiyle bu kısıtlamaları ortadan kaldırarak daha geniř bir geliřtirme ortamı sunmuřtur.

Kivy'nin a ık kaynak yapısı, sade s z dizimi ve hızlı prototipleme imk nı,  zellikle eėitim alanında ve bireysel projelerde tercih edilme nedenlerinden biridir. Ek olarak KivyMD, Plyer ve benzeri yardımcı k t phaneler sayesinde hem tasarım hem de donanım eriřimi konusunda daha zengin ve iřlevsel uygulamalar geliřtirmek m mk n h le gelmektedir. Bu y n yle Kivy, Python geliřtiricilerine yalnızca masa st  deėil, aynı zamanda mobil d nyada da kendini ifade edebilme fırsatı sunar.

Genel olarak deėerlendirildiėinde Kivy, tek bir programlama diliyle farklı platformlara uygulama geliřtirmeyi m mk n kılan yapısıyla sade, esnek ve eriřilebilir bir   z m olarak  ne  ıkmaktadır. Saėladığı bu avantajlar sayesinde, Python ekosisteminde hem  ėrenme hem de uygulama geliřtirme s re lerine ger ek anlamda katkı saėlayan deėerli bir framework h line gelmiřtir.

Kaynakça

Kivy Organization. *Kivy Official Documentation*. <https://kivy.org/doc/stable/> (Eriřim Tarihi: 2025)

Kivy Organization. *Kivy GitHub Repository*. <https://github.com/kivy/kivy> (Eriřim Tarihi: 2025)

KivyMD Developers. *KivyMD: Material Design Components for Kivy*. <https://github.com/kivymd/KivyMD> (Eriřim Tarihi: 2025)

Buildozer Community. *Buildozer Documentation: Packaging Kivy Apps for Android*. <https://buildozer.readthedocs.io/en/latest/> (Eriřim Tarihi: 2025)

Python Software Foundation. *Tkinter — Python Interface to Tcl/Tk*. <https://docs.python.org/3/library/tkinter.html> (Eriřim Tarihi: 2025)

Qt Company. *Qt for Python (PySide) Documentation*. <https://doc.qt.io/qtforpython/> (Eriřim Tarihi: 2025)

Riverbank Computing. *PyQt Documentation*. <https://www.riverbankcomputing.com/software/pyqt/> (Eriřim Tarihi: 2025)

Real Python. *Introduction to Kivy: Cross-Platform Python Framework*. <https://realpython.com/mobile-app-kivy-python/> (Eriřim Tarihi: 2025)

OpenCV Team. *OpenCV Documentation*. <https://docs.opencv.org/> (Eriřim Tarihi: 2025)

Plyer Developers. *Plyer: Platform-Independent API for Hardware Access*. <https://github.com/kivy/plyer> (Eriřim Tarihi: 2025)

