月刊マガラボ

Vol.04

2025.8

中国・杭州の魅力(Click here for the English text)

杭州は、中国東部の浙江省に位置する都市です。歴史遺産と自然の美しさで知られており、中国で最も美しい都市の一つとも言われています。現在、杭州は近代的な都市に成長し、優れた交通システムを備えています。街を歩くと、戸建ての家はほとんど見かけず、人々は一般的にアパートに住んでいます。これはおそらく、人口密度と土地の効率的な利用を促すためでしょう。杭州は交通システムも進んでいます。特に都心部や周辺地域を散策したい場合、多くの路線が市内の様々な場所を結んでいるため、地下鉄は人気があり効率的な交通手段です。バスはより手頃な価格ですが、地下鉄よりも時間がかかるため、選択肢の一つとなります。また、杭州は再生可能エネルギー車の利用を推進した中国初の都市の一つでもあります。そのため、高速道路では多くの電気自動車や電動バイクを見かけるのも不思議ではありません。しかし、音のしない電動バイクは不意を突かれることがあるため、そばを歩く際には注意が必要です。杭州政府は電気自動車の普及を積極的に支援しているようで、公共の場所でも充電インフラを簡単に見つけることができます。杭州は急速な発展を遂げてきましたが、今なおその歴史的・文化的遺産を保持しています。この街の長い歴史と共に、数多くの歴史的建造物が存在します。その一つが、中国で最も古く、最も初期に建てられたモスクの一つである鳳凰寺です。いくつかの資料によると、このイスラム教徒の礼拝所は7世紀頃、唐の時代に建てられたとされています。イスラム教自体は6世紀に預言者ムハンマドによってサウジアラビアのメッカの街でもたらされたことを考えると、これは確かに非常に興味深いことです。したがって、このモスクは、イスラム教がいかに早く地元の人々に受け入れられたかの象徴です。平和を教え、善行をなし、同胞を助け合うというイスラム教の中核的な教えが、世界各地でイスラム教が広まりやすくなった要因です。私は、上城区の中山路の中ほどに位置するこの歴史的なモスクを訪れる機会がありました。地下鉄を使えば、この旧市街エリアにあるモスクへ簡単にアクセスできます。モスク全体の敷地面積は3,300平方メートルで、主要な建物の構造はイスラム建築の様式を保ちつつも、中国の伝統建築の影響を強く受けています。杭州には他にも、ユネスコの世界遺産に登録されている西湖や、南山の麓に位置する中国で最も古く最大級の仏教寺院の一つである霊隠寺、そして伝統工芸品や地元の食べ物が楽しめる河坊街でのグルメ散策など、訪れるべき観光名所がたくさんあり

これから訪れる方へのアドバイスです。Googleマップを含むGoogleとその関連サービスは利用できないため、自国からローミングデータを準備しておけば自由に使えます。もし中国語が話せるのであれば、Baidu（百度）のようなGoogleの代替アプリを使うこともできます。また、多くの地元の人々が英語を話せず、ローマ字も読めないことに驚かないでください。そのため、翻訳アプリも非常に役立ちます。楽しい冒険を！

執筆：Faisal Hadi

留学生コラム

When Humans Fall in Love with Machines: Ethics, Privacy, and Emotional Manipulation in AI Relationship

In an age where artificial intelligence is becoming increasingly personal, a new kind of relationship is emerging—one not between humans, but between humans and machines. Some now confide in AI companions, turning to them for emotional support, friendship, or even romantic connection. For instance, a person grieving a lost loved one might ask an AI to replicate the voice or personality of that person, creating an illusion of continued presence.

While these interactions can offer comfort, they raise profound ethical questions. Are we truly being helped—or just lulled into emotional dependence on a machine that cannot feel? There's a risk that emotionally vulnerable users may develop attachments to AI systems that are designed to respond with empathy but lack genuine understanding. The data collected from such intimate exchanges also raises privacy concerns, as emotional conversations become products to be stored, analyzed, and potentially exploited.

This growing phenomenon challenges traditional ideas of love, connection, and authenticity. As we continue to blur the line between human and machine, society must carefully consider how far emotional AI should go—and whether machines can, or should, be trusted with our hearts.

執筆：Fisilmi Azizah Rahman

先生コラム(Click here for the English text)

Pythonは**インタプリタ型言語**であり，コンパイルなしで直接ソースコードを実行します．このため，スクリプトはビルド作業なしで簡単に配布および実行できるので，迅速な開発やコードの共有に特に便利です．ただし，この利便性には，Pythonインタープリタとそのコードが依存する外部ライブラリやパッケージが必要な**実行時依存関係**であるというトレードオフが伴います．
Pythonで開発する際，実行環境を確実にセットアップし管理できることが重要です．一貫性のない環境では，依存関係の欠落や非互換性のためにコードが失敗する可能性があります．この記事は，Python開発環境の**信頼性が高く，再現可能なセットアップ**を確保するためのガイドとして役立つことを目的としています．
最初の依存関係は，**Pythonインタープリタ**そのものです．Pythonは継続的に開発されており，新機能が導入され，古い機能が廃止されることもあります．あるバージョンのPythonで動作するスクリプトが，構文，標準ライブラリ，または動作の変更により，別のバージョンでは動作しなくなる可能性があります．Pythonのバージョンを明示的に管理することは，信頼性が高く再現可能な開発セットアップを作成するための重要な部分です．プロジェクトによって**異なるバージョンのPython**が必要になる場合があるため，複数のバージョンをインストールして切り替える必要があることもよくあります．pyenv，uv，condaなどのツールを使用すると，これが可能になります．
インタープリタ自体に加えて，ほとんどのPythonプロジェクトは外部ライブラリまたはパッケージに依存しています．これらは，他者によって書かれた追加機能を提供する再利用可能なコードの断片です．これらのパッケージをインストールおよび管理するために，pip，conda，uvなどの**パッケージマネージャー**が使用されます．これらのツールは，Pythonライブラリをホストするオンラインソースであるパッケージリポジトリからパッケージをダウンロード，インストール，更新することを処理します．最も一般的な2つのリポジトリは，PyPI（Python Package Index，公式で最も広く使用されているソース）とAnaconda Repository（主にcondaパッケージマネージャーで使用される）です．一般に，異なるソースからのパッケージを混合したり，同じ環境で複数のパッケージマネージャーを使用したりすることは，競合や不整合につながる可能性があるため避けるべきです．
デフォルトでは，パッケージマネージャーはパッケージをグローバル環境にインストールします．これにより，異なるプロジェクトが同じ依存関係の異なるバージョンを必要とするときに競合が発生する可能性があります．これらの問題を回避するには，仮想環境を使用することが最善の方法です．**仮想環境**は，グローバルインストールとは別の，独自のPythonインタープリタと依存関係を持つ隔離されたワークスペースです．これにより，各プロジェクトは他のプロジェクトに干渉することなく，独自のパッケージセットを維持できます．venv，virtualenv，conda，uvなどの**環境マネージャー**は，環境のセットアップを自動化し，依存関係の隔離を処理するのに役立ちます．
環境マネージャーを使用する場合，依存関係の管理は通常，手動の個別のステップであり，requirements.txtまたはenvironment.ymlファイルを作成および維持することが含まれます．これらには通常，直接的な依存関係のみが含まれます．これは，推移的（間接的）な依存関係は通常ロックされず，インストール中に動的に解決されることを意味します．したがって，**ビルドは完全に決定的ではなく**，わずかに異なる環境や，追跡が困難な微妙なバグにつながる可能性があります．PoetryやuvなどのプロジェクトマネージャーはPythonにおける非決定的なビルドの問題を解決するために，より統合されたアプローチを使用します．
依存関係の管理，環境の作成，およびパッケージングを単一の合理化されたワークフローに組み合わせることで，これらのツールは開発者が単一の構成ファイル（通常はpyproject.toml）でプロジェクトを管理できるようにします．このファイルにプロジェクトの直接的な依存関係が指定されると，ツールはすべての推移的（間接的）な依存関係を自動的に解決し，プロジェクトが正しく実行するために必要な正確なバージョンを見つけます．その後，すべてのパッケージ（直接的および間接的の両方）の特定のバージョンがロックファイルに注意深く記録され，このロックファイルを使用して，どこでもまったく同じ環境を再作成でき，一貫性のある再現可能なビルドを保証します．
常に厳密に必要というわけではありませんが，この記事がPython開発にプロジェクトマネージャーを使用することの利点を明確に示してくれたことを願っています．言及された多くのツールの中で，uvは，Python自体のインストールから，依存関係の管理，仮想環境の作成，プロジェクトのパッケージングまで，すべてを処理するため，最も包括的なツールとして際立っています．したがって，どのツールを採用すべきかわからない場合は，ワークフローを簡素化および合理化するために，プロジェクトマネージャーとしてuvを使用することをお勧めします．uvの使用に関するクイックスタートガイドはこちらにあります．

執筆:Yeoh Wen Liang

就活座談会

先日、奥村研との合同で就活座談会に参加してきました！
M2の学生が就活を開始した時期や活用した就活サイト・メンター、就活のスケジュールなどの就活に関する 情報を1人ずつ黒板に書き出して発表し、そのあとはM1の気になる質問に答えました。不安や疑問を解消できたのではないでしょうか。今年のM1の学生は、例年より早いペースで就活をしており、とてもうまく進んでいる印象があります。
このまま第一志望の企業の内定を勝ち取って欲しいですね！
B4のみなさんも就活で聞きたいことがあればぜひ先輩を頼ってくださいね！

執筆:姫城　太一

編集後記

こんにちは，広報の姫城です．
今回の表紙は、先日開催された筑後川花火大会での1枚です。
みなさんはお盆はゆっくりできましたか？
まだまだ暑い日は続きます。
体調に気をつけてお過ごしください。